Virtuelles Prüfungssystem

Philipp Hügelmeyer, Robert Mertens

1 Einleitung

Klausuren und die damit verbundene Korrekturarbeit binden gerade bei Großveranstaltungen einen nicht unerheblichen Teil der zur Verfügung stehenden Arbeitskraft. Da Großveranstaltungen in den meisten Fällen Veranstaltungen im Grundstudium sind, bleiben die Inhalte oft gleich oder werden nur wenig verändert. Daher können auch Fragen und Übungsaufgaben in vielen Fällen wieder verwendet werden. Da nicht nur Fragen und Aufgaben, sondern auch Lösungsansätze und Antworten oft immer wieder auftreten, lässt sich die Korrektur hier mit vertretbarem Aufwand automatisieren oder zumindest teilweise automatisieren. Dies gilt nicht nur für multiple-choice Aufgaben, sondern auch für Programmieraufgaben und in Einzelfällen sogar für Freitext. Durch die Automatisierung der Korrektur werden Kapazitäten frei, die zur intensiveren Betreuung der Studierenden genutzt werden können. Dies gilt nicht unbedingt für die Lehrenden der jeweiligen Veranstaltungen, die mit der eigentlichen Korrekturarbeit meist ohnehin nicht befasst sind, sondern für wissenschaftliche Mitarbeiter und studentische Hilfskräfte. Daher wird hier nicht Lehr- sondern Betreuungskapazität frei. Dies gilt nicht nur für Klausuren, sondern auch für wöchentliche Übungsaufgaben und Tutorien. In einigen Fällen ist an der Universität Osnabrück die regelmäßige Bearbeitung von Übungsaufgaben erst möglich geworden. In anderen Veranstaltungen ist es durch den Einsatz halbautomatisierter Korrekturprogramme gelungen, höhere Studentenzahlen bei gleich bleibender Personalstärke zu betreuen.

Im weiteren Verlauf werden Erfahrungen geschildert und analysiert, die an der Universität Osnabrück über mehrere Jahre hinweg mit dem Einsatz und der Entwicklung von Systemen zu Korrekturunterstützung bei Tutorien und Klausuren gemacht worden sind.

2 Hintergrund

1997 wurde an der Universität im Rahmen des Projektverbundes ›VIRTUELLER CAMPUS‹ der Universitäten Hannover - Hildesheim - Osnabrück mit der Entwicklung eines intelligenten, tutoriellen Systems und einem prototypischen Lehr-Lern-Modul für die Programmiersprache Prolog begonnen (Wagner, 2000, S. 18). In den Folgejahren wurde ein Lisp-Modul für das System entwickelt (Peylo, Thelen, Rollinger & Gust, 2000, S. 72). Beide Module wurden in mehreren Lehrveranstaltungen

zunehmend erfolgreich eingesetzt. Nach Ablauf des Projektes wurde das Tutor-Programm verschiedenen technischen und didaktischen Anforderungen angepasst und unter dem Namen MVC in ausgewählten Veranstaltungen der Studiengänge Cognitive Science, Betriebswirtschaftslehre, Computerlinguistik und Künstliche Intelligenz eingesetzt (Gust, Hügelmeyer, Mertens & Rollinger, 2003, S. 157). Mittlerweile ist eine Neuimplementation des Systems unter dem Namen ViPS (Virtuelles Prüfungs-System) in die an der Universität Osnabrück universitätsweit eingesetzte Lehr-Lernplattform Stud.IP integriert.

2.1 Möglichkeiten und Einschränkungen

2.1.1 Programmieraufgaben

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 Programmiersprachen Lisp und Prolog implementiert, die Ankopplung anderer Programmiersprachen ist geplant.

Die strukturelle Analyse von Programmiercode ist bei logischen und funktionalen Programmiersprachen relativ gut erforscht und bereits in verschiedenen Systemen implementiert (vgl. Anderson, Boyle, Corbett & Lewis, 1990 und Peylo, Teiken, Rollinger & Gust, 1999). Imperative und objektorientierte Programmiersprachen sind diesbezüglich problematischer, eine strukturelle Analyse ist in ViPS daher nicht geplant.

2.1.2 Lückentexte

Obwohl die automatische Korrektur von Lückentexten auf den ersten Blick sehr einfach erscheint, sind dabei eine Reihe von Faktoren zu beachten. Zunächst stellt die Verwendung von nicht antizipierten Synonymen ein Problem dar. Die Korrektur ist daher halbautomatisiert. Bereits bekannte Wörter werden automatisch klassifiziert, unbekannte Wörter müssen manuell korrigiert werden und werden dann automatisch in die Liste bereits bekannter Lösungen aufgenommen. Ein weiteres Problem stellen Rechtschreibfehler und verschiedenen Schreibweisen von Umlauten dar. Das System verfügt daher über eine Toleranzgrenze für orthographische Abweichungen. Da unterschiedliche Szenarien, wie zum Beispiel auch Fremdsprachentests, denkbar sind, kann diese Toleranzgrenze von Aufgabe zu Aufgabe individuell eingestellt werden. Darüber hinaus ist die Verwendung von regulären Ausdrücken möglich.

2.1.3 Freitext

Freitexte stellen naturgemäß das größte Problem dar, wenn es um automatische Auswertungen geht. Deshalb wird hier mit verschiedenen Ähnlichkeitsmaßen gearbeitet. Zum einen gibt es eine Liste von Wörtern, die im Text vorkommen sollten, und eine Negativliste mit Wörtern, die nicht im Text vorkommen sollten. Des Weiteren kann der Text mit einer Musterlösung verglichen werden, und es werden,

anhand von Hash-Werten über den Text und die Leerzeichen im Text, Ähnlichkeiten zwischen verschiedenen Lösungen bestimmt, so dass diese an Hand solcher Informationen gruppiert werden können. Ähnliche Lösungen können somit von der selben Person korrigiert werden und kopierte können leichter erkannt werden.

2.1.4 Multiple-Choice

Multiple-Choice Aufgaben gelten gemeinhin als die am leichtesten, automatisch auszuwertenden Aufgaben. Nichtsdestotrotz gibt es viele verschiedene Vorstellungen, wie diese Aufgaben bewertet werden sollen.

Im System sind mehrere Typen von Multiple-Choice- und Single-Choice-Aufgaben realisiert. Single-Choice-Aufgaben lassen dabei genau eine Antwort zu, Multiple-Choice auch mehr. Diese beiden Aufgabentypen gibt es jeweils mit und ohne Enthaltung. Durch eine Enthaltung besteht die Möglichkeit auszudrücken, dass man keine Antwort weiß. Abhängig davon, ob die Möglichkeit zur Enhaltung gegeben ist, werden unterschiedliche Bewertungsalgorithmen verwendet. Des Weiteren gibt es im System Ja/Nein-Fragen als Spezialform der Multiple-Choice-Aufgaben. Hierbei kann man genau eine Frage mit ja oder nein beantworten.

Wenn es bei einer Multiple-Choice-Aufgabe die Möglichkeit der Enthaltung gibt, so wird für jede korrekte Anwort ein Rohpunkt addiert, für jede nicht korrekte Anwort ein Rohpunkt subtrahiert. Rohpunkte haben dabei nichts mit den Punkten zu tun, die für eine Aufgabe vergeben werden, da diese von Übungsblatt zu Übungsblatt variieren können, sondern werden nur zur Internen Berechnung genutzt. Enthält sich der Student, bleibt die Anzahl der erreichten Punkte unverändert. Ist keine Enthaltung möglich, so werden nur die Rohpunkte für eine richtig angekreuzte Antwort addiert. Dann wird bei der Möglichkeit zur Enthaltung durch die Anzahl aller Antwortmöglichkeiten, bei der Multiple-Choice-Aufgabe ohne Enthaltung durch die Zahl der richtig beantworteten Fragen, geteilt und dann mit 100 multipliziert, um einen Prozentwert zu erhalten. Dabei ist beabsichtigt, dass die Studierenden bei Aufgaben mit Enthaltung auch negative Punkte erhalten können. Negative Punkte verhindern dabei, dass reines Raten einen Vorteil gegenüber der Enthaltung verschafft. Bei Single-Choice-Aufgaben wird für die korrekte Antwort die volle Prozentzahl gegeben. Wenn Enthaltungen möglich sind, für eine falsche -100 Prozent und sonst immer Null.

2.1.5 Zuordnungsaufgaben

Bei Zuordnungsaufgaben handelt es sich um Aufgaben, bei denen Werte anderen Werten zugeordnet werden sollen. Fragen des Typs »Ordnen Sie folgende Politiker ihrer Partei zu!« fallen in diese Kategorie. Auch Reihenfolge-Aufgaben sind hier zu nennen.

Augenblicklich ist dieser Typ nicht im System implementiert, die zugrunde liegende Bewertungslogik ist jedoch vergleichsweise einfach, so dass dieser Aufgabentyp bei Bedarf schnell eingebunden werden kann.

2.2 Anwendungsziel des Szenarios

Anwendungsziel ist es, Routineaufgaben von Tutoren und Übungsleitern zu automatisieren, um so mehr Studierende mit gleich bleibendem Personalaufwand betreuen zu können und mehr Kapazität für die Individualbetreuung der Studierenden zu schaffen. Durch integrierte Überblicks- und Auswertungsfunktionen lässt sich schneller und einfacher ein Leistungsüberblick über den gesamten Kurs verschaffen. Somit können problematische Bereiche innerhalb des Lernstoffes schneller lokalisiert und die Lehre besser dem Niveau der Studierenden angepasst werden.

2.3 Technisches Konzept

Das System ist in seiner augenblicklichen Implementation als PHP-basierte Client-Server-Lösung realisiert. Es stellt damit sehr geringe Anforderungen an die Rechner, an denen die Studierenden arbeiten. Diese Strategie hat sich gegenüber früheren Ansätzen vor allem durch eine erhöhte Benutzerakzeptanz abgehoben, da technische Probleme auf Benutzerseite praktisch ausgeschlossen sind (vgl. Peylo & Teiken, 1998 sowie Gust, Hügelmeyer, Mertens & Rollinger, 2003). Durch die Einbindung des Systems als Modul in die Lehr-/Lernplattform Stud.IP sind Nutzerverwaltung und sonstige administrative Aspekte ausgelagert und müssen nicht doppelt bearbeitet werden.

2.4 Zielgruppe und Teilnehmer

Zur Betreuung von Übungsgruppen und Tutorien kann das System in Lehrveranstaltungen beliebiger Größe eingesetzt werden. Problematisch wird der Einsatz bei Klausuren in größeren Lehrveranstaltungen. Die hierbei auftretenden Probleme sind allerdings eher organisatorischer Natur. Da Rechnerräume meist nur kleineren Gruppen Platz bieten, müssen Klausuren entweder in mehreren Räumen gleichzeitig oder zeitversetzt stattfinden. Bei zeitversetzt stattfindenden Klausuren muss mit Kommunikation der Studierenden über den Inhalt der Klausur gerechnet werden. Das gegenseitige Abschreiben der Studierenden während der Klausur kann durch die Verwendung von Aufgabenpools mit zufälliger Auswahl von Aufgaben derselben Kategorie und mit zufälliger Reihenfolge der Aufgaben erschwert werden. Dadurch ist es möglich, die Kapazität der Rechnerräume besser zu nutzen und auch direkt nebeneinander stehende Rechner zu verwenden. Thematisch ist der Einsatz des Systems durch die bereits beschriebenen Aufgabentypen beschränkt. Am besten eignet es sich für den Einsatz in Programmierkursen. Es ist jedoch auch für Veranstaltungen anderer Thematik ohne viel Aufwand möglich, Wissenstests aus den vorhandenen Aufgabentypen zusammenzustellen.

3 Organisatorische und didaktische Aspekte

3.1 Veranstaltungsform

Das System wurde an der Universität Osnabrück bisher in Vorlesungen und begleitenden Übungen/Tutorien eingesetzt. Dabei wurde die Anwendung sowohl für die Abwicklung wöchentlicher Übungsaufgaben und von Klausuren als auch als Selbstlernsystem genutzt, in dem Studierende sofort ein Feedback über ihre Lösung erhalten. Der Einsatz in Projekten oder Seminaren bietet sich kaum an, da ViPS vornehmlich zum Abfragen und Testen von Wissen und Fähigkeiten konzipiert ist. In Seminaren und Projekten dagegen ist die eigenständige und größtenteils komplexe Problembearbeitung das Ziel der Lehrveranstaltung.

3.2 Organisation

ViPS und seine Vorgängersysteme wurden in mehreren Vorlesungen mit jährlichem Fortschreibungsrhythmus eingesetzt (Peylo, Thelen, Rollinger & Gust, 2000 sowie Gust, Hügelmeyer, Mertens & Rollinger, 2003). In den meisten Veranstaltungen wurden sowohl wöchentliche Übungsaufgaben als auch ein bis zwei Klausuren mit dem System durchgeführt. Die Anwendung wurde dabei wiederholt in den Veranstaltungen eingesetzt, so dass Aufgaben aus den vorangegangenen Semestern wieder verwendet werden konnten.

3.3 Sozialform

Zur Bearbeitung der Übungsaufgaben können die Studierenden in Kleingruppen von zwei bis drei Personen eingeteilt werden. Es gibt dabei zwei Typen von Übungsgruppen. In manchen Gruppen teilen sich die Studierenden die Aufgaben und bearbeiten sie einzeln, in anderen Gruppen werden die Aufgaben gemeinschaftlich bearbeitet. Im Tutorium werden die Aufgaben mit einer Gruppe von 15-35 Studierenden besprochen. Jede Gruppe wird von einem Tutor betreut, der für die Korrektur der Aufgaben zuständig ist und über das System, genau wie die Gruppe selbst, Kommentare zu den Aufgaben abgeben kann. Die Bearbeitung der Klausuraufgaben erfolgt klassisch einzeln. Auch wird die Korrektur der Aufgaben nicht zwingend von dem zugeordneten Tutor erledigt.

3.4 Wiederverwendbarkeit der Daten

In vielen Grundstudiumsveranstaltungen verändert sich der Lehrstoff nur sehr langsam. Viele Aufgaben können daher über Jahre genutzt und wieder verwendet werden. Bei Klausuraufgaben ist auf Variationen der Aufgabenstellung zu achten. Bei Übungsaufgaben jedoch ist das Ziel der Aufgaben nicht nur die Benotung der Studierenden, sondern vorrangig die Möglichkeit das Gelernte anzuwenden und zu

wiederholen. Übungsaufgaben können daher ohne Probleme wieder verwendet werden. Da alle Aufgaben im System gespeichert werden, ist es außerdem möglich, über mehrere Semester hinweg einen Pool von Aufgaben aufzubauen. In diesem Pool kann anhand verschiedener Kriterien oder im Volltext gesucht werden. Kriterien sind die Voraussetzungen und die behandelten Themen. Außerdem kann jeder, der Aufgaben erstellt, entscheiden, ob er diese veröffentlichen möchte, sodass auch andere Dozenten diese Aufgaben in ihren Kursen verwenden können.

4 Technische Aspekte

4.1 Technische Voraussetzungen

Als Server für ViPS kann ein beliebiger Stud.IP (http://www.studip.de) Server dienen. Das System sollte über einen Webserver (z.B. Apache) mit PHP4 Unterstützung verfügen. Als Backend wird eine MySQL-Datenbank verwendet. Darin wird für das Virtuelle Prüfungssystem eine eigene Datenbank angelegt, damit es nicht zu Konflikten mit dem eigentlichen Stud.IP-System kommen kann. An zusätzlichen Paketen zu einer normalen Stud.IP-Installation wird das PHP4-domxml-Modul benötigt, da alle Aufgaben intern als XML repräsentiert sind. Zusätzlich gibt es für Prolog-Aufgaben einen in Perl realisierten Server, der auf einem vom Webserver getrennten System läuft und mit Hilfe einiger Shellskripte und dem SWI-Prolog-Interpreter die Auswertung von Prologaufgaben übernimmt. Auch andere angedachte Erweiterungen wie die Anbindung von MATLAB werden über solche Serverlösungen realisiert, um die Anforderungen an den eigentlichen Webserver so gering wie möglich zu halten.

4.2 Übersicht über das System

Der Nutzer steuert die Anwendung über eine Webschnittstelle. Die Funktionen sind in Form eines Karteikastens mit mehreren Reitern dargestellt, auf denen je nach Status im System verschiedene Optionen zur Verfügung gestellt werden. So kann der Dozent in der Gruppenverwaltung die Studenten zu Gruppen zusammenfassen und die Tutoren den Gruppen zuordnen. Ein Studierender kann je nach Konfiguration nur sich selber zuordnen oder auch nur seine Punkte einsehen. Darüber hinaus gibt es weitere Reiter für Übungsaufgaben, Klausuren und eine Übersicht über die Lösungen. Für den Dozenten gibt es einen Administrationsbereich, in dem wichtige Grundeinstellungen für das Systemverhalten vorgenommen werden können. Auch kann der Dozent Aufgaben mit einem einfachen Texteditor in einer einfachen Syntax formulieren und über den Browser ins System hochladen. Dort werden die Aufgaben dann in die interne Repräsentation konvertiert. Alle auf diese Art ins System importierten Aufgaben tauchen dann im Aufgabenpool des Dozenten auf.

4.3 Schulungsbedarf

Der Schulungsbedarf für die Bedienung des Systems ist relativ gering, da die Module zur Aufgabenerstellung klar und einfach strukturiert sind. Bedienelemente und Konzepte sind an Stud.IP angepasst. Wer Stud.IP kennt, wird sich auch in ViPS gut und schnell zurecht finden.

Weitaus wichtiger ist es, mit den Dozenten zu erarbeiten, wie sich Übungsaufgaben und Klausuren in die Lehrveranstaltungen integrieren lassen, und mit welcher Aufgabenstellung welches Wissen abgefragt werden kann. Dies ist jedoch stark abhängig von der Lehrkultur, die ein Dozent oder ein Fach vorher gelebt hat. Wenn schon vor der Einführung eines Übungs- und Prüfungssystems Testate oder wöchentliche Übungszettel üblich waren, fällt es den Dozenten naturgemäß leichter, diese Fragestellungen in ein elektronisches System zu übertragen.

4.4 Softwareentwicklung

Die benötigte Software wurde von Mitarbeitern des Zentrums virtUOS entworfen und entwickelt. Dabei konnte auf die Erfahrungen aus dem Virtuellen Campus Projekt (Peylo, Thelen, Rollinger & Gust, 2000) zurückgegriffen werden. Der Code wurde aber vollständig neu implementiert. Da sowohl Stud.IP als auch das hier beschriebene Projekt unter der GNU Public License (Free Software Foundation, 2004) stehen, konnten viele Komponenten und Strukturen aus Stud.IP übernommen oder an Stud.IP angelehnt werden. Beispiele hierfür sind die Nutzerauthentifikation, die Rechtestufen und die Gruppenverwaltung. Die verschiedenen Aufgaben wurden als Objekte realisiert und können im laufenden Betrieb in das System integriert werden. Datenbankänderungen o.ä. sind dafür nicht notwendig. Natürlich wird für die Auswertung von Programmiersprachen und mathematischen Aufgaben auf bestehende Software zurückgegriffen, die über einen Wrapper mit den Aufgaben und Auswertungsalgorithmen kommuniziert. So wird zum Beispiel für die Kurse im Fach Cognitive Science SWI-Prolog als Interpreter benutzt.

5 Bewertung

Das System bietet für viele Veranstaltungsformen abgesehen vom reinen Zeitgewinn auch neue Konzepte zur Wissenskontrolle und wichtiges Feedback für den Dozenten, der seine Lehre anhand dieser Leistungskontrolle verbessern kann. Das Projekt wurde sowohl von den beteiligten Dozenten als auch von den beteiligten Studierenden gut angenommen. Als besonders wichtiges Akzeptanzkriterium hat sich dabei gezeigt, dass das System bei der Einführung problemlos funktioniert und Schwachstellen und Benutzerwünsche sofort verbessert bzw. umgesetzt werden.

5.1 Akzeptanz der Teilnehmer

Technische Probleme mit einem Online-Prüfungssystem führten in der Anfangsphase oft zu geringer Akzeptanz seitens der Teilnehmer (Peylo & Teiken, 1998).

Mit zunehmender Stabilität des Systems verbesserte sich allerdings im weiteren Verlauf des Einsatzes die Akzeptanz. Ein interessantes Phänomen war beim Einsatz von Programmieraufgaben zu beobachten. Es hat sich gezeigt, dass in den Programmiersprachen fortgeschrittenere Studierende die Aufgaben oft online in einer herkömmlichen Programmierumgebung bearbeitet haben, wogegen weniger versierte Studierende während des gesamten Kurses das System verwendet haben (Peylo, Thelen, Rollinger & Gust, 2000, S. 77). Insgesamt hat sich das System inzwischen in mehreren Kursen fest etabliert und wird von den Tutoren und Studierenden als selbstverständlich angenommen. Ein Vorteil bei der Einführung von ViPS war sicherlich, dass die Lehrenden und Studierenden durch den Einsatz von Stud.IP schon an die Bedienoberfläche gewöhnt waren und sich so innerhalb des virtuellen Prüfungssystems schnell zurechtgefunden haben. Dennoch hat es sich bewährt, vor einer Klausur eine Probeklausur zu schreiben, damit auch diejenigen, die sonst wenig mit Computern umgehen oder Stud.IP kaum benutzen, sich auf das System einstellen können, damit nicht die Performanz des Kursteilnehmers bei der Bedienung einer Lernplattform, sondern das fachliche Wissen des Einzelnen bewertet werden kann.

5.2 Probleme

Auch durch intensives Testen der Software im Vorfeld des Einsatzes lässt sich Fehlverhalten in unvorhergesehenen Situationen nie ganz ausschließen. Natürlich läuft solch ein Projekt nicht ganz reibungslos ab, egal wie viel man vorher testet. Als Entwickler der Software ist man natürlich für viele Usabilitytests ungeeignet, da man das System oft an die eigenen Bedürfnisse anpasst, mit der guten Kenntnis des Systems aber nicht mehr abschätzen kann, ob dieses für einen Außenstehenden auch genau so einfach zu benutzen ist. Deswegen wurde das Produkt zum Einsatz gebracht, sobald die generelle Performanz zufriedenstellend war.

Dies hat sich als sehr gut erwiesen, denn zum einen steigt durch das Benutzerfeedback die Motivation bei der Entwicklung, zum anderen lässt sich die Software so besser an die Bedürfnisse der Anwender anpassen. So wurden durch das Feedback klar, das viele Aspekte vom Nutzer anders gewünscht werden als vom Entwickler erwartet. Wichtig für die Zufriedenheit der Nutzer ist es, das aufgetretene Schwachstellen und Wünsche sofort umgesetzt werden.

5.3 Alternativen

Als Alternativen gibt es auch in anderen (kommerziellen) Lehr-/Lernplattformen wie WebCT oder Blackboard Übungs- und Prüfungssysteme. WebCT ist eine Lehr-/Lernplattform, mit der Kurse verwaltet, Online-Kursmaterial zur Verfügung gestellt werden kann und in der Werkzeuge zur kursinternen Kommunikation bereitgestellt werden. Darüber hinaus beinhaltet WebCT eine so genannte Quiz-Funktion. Im Rahmen dieser Quiz-Funktion bietet WebCT vier verschiedene Aufgabentypen:

- Multiple Choice

- Matching
- Fragen mit einer möglichen Antwort
- Rechenaufgaben

Bei Matching-Aufgaben handelt es sich um einfache Zuordnungsaufgaben. Die automatische Bewertung dieses Aufgabentypes und der Multiple-Choice-Aufgaben ist aufgrund der binären Struktur ihrer Antworten sehr einfach realisierbar. Bei Fragen mit einer möglichen Antwort handelt es sich bezüglich der Bewertung um denselben Typ wie bei Lückentexten im ViPS. Die Auswertung ist daher ähnlich gehalten. Wie ViPS kann auch WebCT mit Synonymen umgehen. Rechtschreibfehler kann WebCT jedoch nur dann ignorieren, wenn diese zuvor vom Lehrenden explizit angegeben werden.

Rechenaufgaben verlangen die Eingabe einer Formel unter Verwendung von Variablen. Diese Variablen initialisiert das System zufällig und vermeidet so, jedem Studierenden dieselbe Aufgabe zu stellen. Bei einem Aufruf lautet die Frage dann z.B. »Wie viel ist 2 + 3« und beim nächsten wird »Wie viel ist 5+1« gefragt. Die Antwort steht aufgrund der Formel bereits zu dem Zeitpunkt fest, an dem das System die Aufgabe stellt. Die Bewertungslogik dieses Aufgabentyps entspricht daher einer Lückentextaufgabe ohne Berücksichtigung von Rechtschreibfehlern. Blackboard verfolgt eine ähnliche Zielsetzung wie WebCT. Auch dieses System verfügt über eine Testkomponente mit automatischer Aufgabenbewertung. Das Learning Management System bietet dabei eine ganze Reihe von unterschiedlichen, automatisch auswertbaren Aufgabentypen, die jedoch allesamt keine freie Eingabe erlauben. Freitextaufgaben sind möglich, müssen aber zu 100 Prozent von Hand ausgewertet werden. Hinsichtlich der Bewertungslogik sind diese Aufgabentypen daher mit einem einfachen Typ wie dem der Multiple-Choice-Aufgaben in ViPS vergleichbar. Hinsichtlich der Komplexität der zur Verfügung stehenden Aufgabentypen hat ViPS durch die Möglichkeit, Programmieraufgaben zu stellen, einen klaren Vorteil. Auch der Lückentextalgorithmus stellt hier einen Pluspunkt dar, der in einem der beiden Referenzsysteme auch implementiert ist.

Von allen betrachteten Aufgabentypen sind allerdings Programmieraufgaben oder Aufgaben mit großer Texteingabe noch nicht adäquat automatisch korrigierbar. Mit diesen ist es jedoch möglich, Transferleistungen statt bloßem Faktenwissen abzufragen.

Weiterführende Informationen zu beiden Systemen sowie eine Übersicht über weitere Lehr-/Lernplattformen sind bei Baumgartner, Häfele und Maier-Häfele (Baumgartner, Häfele & Mair-Häfele, 2002) zu finden.

5.4 Übertragbarkeit

Ein virtuelles Prüfungssystem wie ViPS ist nicht an bestimmte Fächer gebunden, jedoch lassen sich einige Fachinhalte leichter automatisch korrigieren als andere. So stellt die automatische Korrektur von langen Texten, bei denen nicht die Verwendung von bestimmten Begriffen, sondern ein Argumentationsstrang im Vordergrund steht, oder das automatische Bewerten von Zeichnungen noch zu hohe

Anforderungen an das System. Für andere Testsituationen ist das System jedoch unabhängig von dem abgefragten Wissen geeignet. Auch ist die Anwendung durch das Plugin-System und die Möglichkeit, bei vielen Testsitutationen die Auswertung entsprechend der Fragestellung zu gestalten, das System an den zu erlernenden Stoff leicht anzupassen.

5.5 Curriculare Einbindung

In vielen Fächern wie Informatik, Cognitive Science oder Philosophie ist es üblich, dass zum Erlangen eines Scheins wöchentliche Übungsaufgaben zu lösen sind und am Ende oder auch während des Semesters Klausuren abgelegt werden. Wenn diese Kultur ohnehin schon vorhanden ist, ist es relativ einfach, die Veranstaltung durch ein virtuelles Prüfungssystem zu ergänzen.

In Fächern, in denen wöchentliche Übungsaufgaben nicht üblich sind und Leistungen im Normalfall nicht durch Klausuren oder nur mit Freitexten abgeprüft werden, stellt sich die Einbindung in das Lehrangebot etwas schwierig dar. In diesen Fächern bietet es sich an, das virtuelle Prüfungssystem zuerst für die Studierenden zur Selbstkontrolle ihres Wissens einzuführen. Dies ist zum Beispiel als Vorbereitung auf eine Klausur gut möglich. Später kann die Anwendung, wenn sich Dozenten und Studierende sich daran gewöhnt haben, dann auch zur Leistungsüberprüfung in der Veranstaltung dienen.

5.6 Reifegrad

Obwohl das System ständig weiterentwickelt wird, wurde es bereits in mehreren Veranstaltungen eingesetzt. Dieses Vorgehen ist hilfreich, um das System an die Bedürfnisse der Anwender anzupassen. Nur so erhält man eine Anwendung, die auch von den Benutzern angenommen wird. Es wurden z.B. in der Veranstaltung »Introduction to Artificial Intelligence« alle Übungsaufgaben und Klausuren komplett über das System abgewickelt. Der Einsatz war erfolgreich und erbrachte gleichzeitig Verbesserungsvorschläge für die Usability, um gerade für den Dozenten verschiedene Abläufe zu beschleunigen und andere Sichten auf die erhobenen Daten zu gewährleisten.

Das System wird in den kommenden Semestern in einer Vielzahl von Veranstaltungen für die Lehre eingesetzt. Dies bedeutet jedoch nicht, dass die Entwicklung schon abgeschlossen ist. Durch das Feedback aus den Veranstaltungen wird das System stetig weiter verbessert.

6 Fazit

ViPS ist ein vollwertiges, virtuelles Prüfungssystem, das aus einem langjährigen Erfahrungsschatz schöpfen kann und mit der Zeit an den Bedürfnissen der Benutzer und den Anforderungen aus dem praktischen Einsatz gewachsen ist. Es hat sich als stabil, robust und flexibel bewährt. Als ein besonderer Vorteil hat sich bei

Programmieraufgaben das direkte Feedback durch das System erwiesen. Auch die Reduzierung des Korrekturaufwandes ist für die Lehrenden attraktiv, obwohl die Aufgaben zunächst für das System aufbereitet werden müssen. Es ist davon auszugehen, dass sich die Nutzerzahl in den nächsten Jahren massiv erhöhen wird und dass das System durch stetige Weiterentwicklung immer attraktiver wird.

7 Literatur

Anderson J. R., Boyle C. F., Corbett, A. T. & Lewis, M. W. (1990). Cognitive modeling and intelligent tutoring. *Artificial Intelligence*, 42 (1), 7-51.

Baumgartner, P., Häfele, H. & Mair-Häfele K. (2002). *E-Learning Praxishandbuch, Auswahl von Lernplattformen. Marktübersicht - Funktionen - Fachbegriffe.* Insbruck: StudienVerlag.

Gust, H., Hügelmeyer P., Mertens, R. & Rollinger, C. (2003). Automatische Abwicklung von Klausuren mit dem MVC. In V. Dötsch, G. Schade & K. Hering, *e-Learning and beyond. Proceedings of the 2nd Workshop on e-Learning 2003* (S. 157-165). Leipzig: Hochschule für Technik, Wirtschaft und Kultur, Fachbereich Informatik, Mathematik und Naturwissenschaften.

Peylo, C., Teiken, W., Rollinger, C. & Gust, H. (1999). Der VCProlog-Tutor, eine Internet-basierte Lernumgebung. *Künstliche Intelligenz (4/99)*, 32-36.

Peylo, C. & Teiken, W. (1998). *Ein internetbasiertes tutorielles System als Java-Applet. Erfahrungen mit einer plattformunabhängigen Sprache unter verschiedenen Plattformen.* Universität Osnabrück.

Peylo, C., Thelen, T., Rollinger, C. & Gust, H. (2000). 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.* http://tobiasthelen.de/doc/cpeylo_tthelen_crollinger_hgust_its-2000.pdf

Wagner, E. (Hrsg.) (2000). *Projektverbund »VIRTUELLER CAMPUS« der Universitäten Hannover - Hildesheim - Osnabrück (Projekt VC I: 1997 - 2000) Abschlussbericht.* http://www.uni-hildesheim.de/ZFW/vc/vc-bericht.pdf (9.9.2004).

Free Software Foundation. (2004). *GNU General Public License.* http://www.gnu.org/copyleft/gpl.html (9.9.2004).