Multimediagestütztes Lehren im Hochschulbereich – Möglichkeiten in der Lehrerausbildung

Abschlußbericht des MALL-Projekts

Andreas Kerren, Reinhard Wilhelm und Stephan Diehl
Universität des Saarlandes, FR 6.2 Informatik
Postfach 15 11 50, D-66041 Saarbrücken
{kerren,wilhelm,diehl}@cs.uni-sb.de

20. Juni 2000

1 Einführung

Im folgenden untersuchen wir die Frage, ob Lehrer in die Lage versetzt werden können, generische Lehr- und Lernsysteme für Unterrichtseinheiten sinnvoll anzupassen. Wir beginnen mit einer Begriffsbestimmung. Dann folgt eine Bestandsaufnahme relevanter Systeme, wobei wir zwei Systeme exemplarisch näher betrachten. Aufgrund der in der Bestandsaufnahme gewonnenen Informationen diskutieren wir die Ausgangsfrage in Abschnitt 4.

2 Begriffsbestimmung: Generische und generative Lehr- oder Lernsoftware

Generische Lehr- oder Lernsoftware wird für einen Unterrichtsbereich erstellt und dann vom Lehrer für spezielle Lehr- bzw. Lerneinheiten instanziiert.

Damit eine Aufgabe *generisch* angegangen werden kann, muss erst einmal klar sein, wovon man abstrahieren kann, und was als gemeinsame Bestandteile aller Instanzen erhalten bleiben muss. Im Falle von Lernsoftware könnte die Lernzielkontrolle gemeinsamer Bestandteil von Systemen quer über viele Fächer sein. Andere Systemteile wie etwa Bedienoberfläche, Schnittstellen zu Wörterbüchern oder anderen Wissensquellen werden fachspezifisch sein. Schließlich bleiben die Elemente übrig, die spezifisch für ein Unterrichtsgebiet sind. Man denke etwa an Software zum Erlernen von Fremdsprachen.

Bei der Konzeption von MALL schwebte uns das folgende Szenario aus dem Bereich Stadtentwicklung vor: Es gibt mehrere mittelalterliche Systeme von Regeln, die die Entwicklung einer Stadt beschreiben, etwa die *Gromatici Veteres*, nach denen Freiburg im Breisgau entworfen wurde. (Der Römer Vitruvius hatte schon früher einen ähnlichen Ansatz verfolgt.) Ein solches Regelsystem könnte in eine generische Software umgesetzt werden, die dann vom Lehrer und evtl. auch den Schülern für die eigenen Stadt instanziiert würde.

Ein etwas weitergehender Ansatz ist es *generative* Systeme zu verwenden, d.h. Systeme, die Software automatisch aus Spezifikationen erzeugen. In Lehr-/Lernsystemen existieren z.B. immer die gleichen Arten von Übungen, d.h. dort bietet sich ein guter Ansatzpunkt für den Einsatz generativer Tools. Nimmt man etwa einen Lückentexteditor, dann könnte ein generatives System Wörter als Lücke markieren, interne Links zu einer entsprechenden Wortdatenbank setzen, oder könnte gar automatisch die Lücken erzeugen, indem z.B. alle Dativ/Akkusativfälle automatisch herausgefiltert und durch Lücken ersetzt werden. In Zusammenhang mit den in Abschnitt 1 genannten Fragestellungen könnte auch eine Lernzielkontrolle miterzeugt werden. Dabei gilt sowohl für generische, als auch generative Systeme, daß sowohl der Lehrer als auch der Lerner solche Systeme verwenden kann [6], wie in Abschnitt 3.4 dargestellt wird. Generische und generative

Systeme erhöhen den Grad an Wiederverwendbarkeit und reduzieren damit Entwicklungskosten, die gerade im Bereich von Lernsoftware sehr hoch sind. Damit ein solcher generischer Ansatz erfolgreich sein kann, müssen mehrere Voraussetzungen erfüllt sein:

- Lehrer der betroffenen Fachrichtungen müssen ein Mindestmaß an Informatikkonzepten beherrschen.
- Die generischen Softwaresysteme müssen mit leicht zu erlernenden Spezifikationsprachen ausgestattet sein, die das Erstellen einer Instanz für einen so vorgebildeten Lehrer möglich machen. Dabei kann eine solche Spezifikationssprache sowohl textuell, als auch graphisch sein, etwa in der Gestalt eines Formulars, etc..
- Die implementierten Systeme müssen den Spezifizierer bei der Fehlersuche unterstützen.

In der MALL-Projektstudie sollte deshalb der Frage nachgegangen werden: Können Lehrer in die Lage versetzt werden, generische Lehr- und Lernsysteme für Unterrichtseinheiten sinnvoll anzupassen? Dazu wurde nach existierenden generischen Lehr- und Lernsystemen recherchiert.

3 Bestandsaufnahme: Generische Lehr- und Lernsoftware

Um einen Überblick über relevante Systeme zu erhalten, wurden umfangreiche Literatur- und Webrecherchen betrieben. Dazu war insbesondere wichtig, systematisch eine große Zahl von Suchbegriffen zu erstellen. Außerdem haben wir Mitarbeiter eines führenden Verlages befragt, der Lehr- und Lernmittel und insbesondere Lernsoftware erstellt. Trotzdem erheben wir keinen Anspruch auf Vollständigkeit, dies wäre in Anbetracht der Diversität und Unübersichtlichkeit der Anbieter von Lehr-/Lernsoftware auch Augenwischerei.

3.1 Von Verlagen verwendete Werkzeuge

Verlage verwenden die üblichen Autorensysteme wie Macromedia Director und Asymetrix ToolBook für einen Großteil der Produktion von Lehr-/Lernsoftware. In Ausnahmefällen wird auch C++ oder eine andere Hochsprache für deren Prgrammierung verwendet. Im Zuge der verstärkten Onlineaktivitäten der Verlage werden auch vermehrt HTML-Editoren, Java-Applikationen und etwa Macromedia Flash eingesetzt. Zunächst geben wir eine kurze Beschreibung und Klassifikation von Autorensystemen und diskutieren anschließend deren Verwendbarkeit durch Lehrer.

3.1.1 Autorensysteme

Fünfstück et al [7] beschreiben Autorensysteme wie folgt: „Autorensysteme ermöglichen die Entwicklung multimedialer Anwendungen in einer grafisch-interaktiven Form unter weitgehendem Verzicht auf die Programmierung, verwenden hierzu visuelle *interface development tools* sowie Werkzeuge zur Erstellung und Bearbeitung von Medien. Sie gehen vom Entwurfsprozeß der Benutzungsoberfläche aus und unterstützen die anderen Phasen des Entwicklungsprozesses häufig nur unvollständig. Die Vorgehensweisen sind weniger an herkömmlichen Programmiertechniken ausgerichtet, sondern beruhen auf einprägsamen Metaphern und einem leichten Zugang der an Gestaltung und Inhalten orientierten Entwickler (MM-Autoren, Designer)“. Insofern scheinen Autorensysteme auf den ersten Blick auch für Lehrer interessante Entwicklungstools zu sein, siehe 3.1.2. Weiterhin gilt für die zu implementierende Lernsoftware: „Die mit Autorensystemen entwickelten Anwendungen weisen sowohl programmbestimmte als auch dokumentenbestimmte Aspekte auf; eine klare Zuordnung zu einem der beiden Typen wird zunehmend schwerer. Das Ergebnis des Entwicklungsprozesses ist i.allg. eine *Beschreibung* in einem zumeist nicht offengelegten Format, die Programmteile in Form von Skripten beinhaltet und von einem Laufzeitsystem interpretiert wird. Ein wesentliches Unterscheidungsmerkmal zu Multimedia Entwicklungsumgebungen

bzw. Multimedia Frameworks ist die objektbasierte Entwicklung, d.h. der Entwickler kommt bei der Programmierung nicht mit Klassen bzw. Komponenten, sondern nur mit deren Instanzen, den Objekten, in Berührung. Autorensysteme stellen zwar Komponenten zur Verfügung, ihre Wiederverwendung geschieht aber ausschließlich in Form instanziierter Objekte, deren Verhalten mit Skripten programmiert wird. Eine Vererbung ist, wenn überhaupt, nur auf Objektebene möglich.“. Autorensysteme entsprechen verschiedenen Paradigmen, von denen eine gängige Auswahl in folgenden Zitaten aus [11, 3] kurz erläutert werden.

Frame-basierte Autorenwerkzeuge „Frame-basierte Autorenwerkzeuge lassen sich dadurch charakterisieren, daß die zu präsentierenden Medienobjekte auf Flächen gelegt werden, die als Karten, Seiten oder auch Dias bezeichnet werden und die im Prinzip einen Bildschirm widerspiegeln, wie ihn ein Benutzer während der Präsentation für einen bestimmten Zeitraum zu sehen bekommt. Eine komplette multimediale Präsentationsanwendung setzt sich aus einer Menge solcher Flächen zusammen, die einem Benutzer in einer bestimmten von ihm durch Interaktionen beeinflußbaren Reihenfolge gezeigt werden. Die Menge der Medienobjekte einer Karte bildet dabei ein komplexes Medienobjekt. Navigationsinteraktionen bewirken im allgemeinen einen Wechsel der gerade präsentierten Karte oder Seite“. Beispiele für Frame-basierte Autorensysteme sind HyperCard von HyperActive Software [13], Asymetrix ToolBook [14], ...

Timeline-basierte Autorenwerkzeuge „Timeline-basierte Autorenwerkzeuge sind dadurch charakterisiert, daß die Medienobjekte auf einer Zeitachse angeordnet werden, die den zeitlichen Verlauf der Präsentation widerspiegelt. Navigationsinteraktionen bewirken im allgemeinen einen Zeitsprung.“ Beispiele für Timeline-basierte Autorenwerkzeuge sind Macromedia Director [15], Macromedia Flash [16], ...

Flowchart-basierte Autorenwerkzeuge „Flowchart-basierte Autorenwerkzeuge sind dadurch gekennzeichnet, daß die Medienobjekte – durch Ikonen bzw. Miniaturen repräsentiert – in Diagrammen durch Kanten miteinander verbunden werden, die den möglichen Verlauf der Präsentation widerspiegeln. Gehen dabei von einem Objekt mehrere Kanten zu unterschiedlichen Objekten aus, so wird die während der Präsentation tatsächlich „durchflossene“ Kante im allgemeinen durch eine Navigationsinteraktion bestimmt. Strukturierte flowchart-basierte Autorenwerkzeuge lassen sich dadurch charakterisieren, daß eine (hierarchische) Strukturierung der Diagramme möglich ist, d.h. bestimmte logisch bzw. funktional zusammenhängende Teile können zusammengefaßt und in ein externes Diagramm ausgelagert werden. Sie bilden ein komplexes Medienobjekt. Das externe Diagramm wird durch ein spezielles Ikon in seinem „Vater-Diagramm“ repräsentiert. Authorware Professional [17] der Firma Macromedia ist das bekannteste strukturierte flowchart-basierte Autorenwerkzeug.“

3.1.2 Verwendbarkeit von Autorensystemen durch Lehrer

Für die Erstellung sehr einfacher Lehr- und Lernsoftware sind Autorensysteme durch Lehrer in akzeptabler Einarbeitungszeit in das Autorensystem zu verwenden. Durch ihre teilweise intuitive Bedienung und graphischer Benutzerschnittstellen bieten sie komfortable Möglichkeiten, einfache CBT-Systeme zu erzeugen. Allerdings sind die führenden Autorensysteme wie ToolBook und Director auch sehr komplex und erfordern teilweise eine gründliche Einarbeitung in die Bedienung und Programmierung meist eingebauter, systemspezifischer Scriptsprachen, wie etwa OpenScript bei ToolBook. Werden komplexere Lernsysteme verlangt, so kommt man mit der Programmierung durch diese eingebauten Scriptsprachen nicht weiter, weil sie meist keine Konzepte wie Modularität, Wiederverwendung von Komponenten, Objektorientierung, o.ä. besitzen. Außerdem werden die damit implementierten Programme interpretiert, was oftmals sehr hohe Laufzeiten zur Folge hat, z.B. wenn man einen Graphlayoutalgorithmus implementieren muß. Hier ist dann die Erstellung von Programmen notwendig, die in Hochsprachen wie C++, Java, etc. programmiert und kompiliert wurden und über geeignete Schnittstellen mit der eigentlichen Autorensystemanwendung kommunizieren, siehe z.B. [8].

3.2 Überblick über relevante generische Systeme

Bevor wir zwei Systeme näher betrachten, geben wir eine kurze Übersicht und grobe Klassifikation relevanter generischer Systeme. Unsere Suche ergab bisher u.a., daß es nur wenige solcher generischer Lehr- und Lernsoftware im Netz gibt. Ferner weisen die folgenden Beispielsysteme unterschiedliche Grade an Generizität auf. Die Liste wird im Abschlußbericht vervollständigt.

Kollektionen von Lernsoftware allgemein

- http://www.sodis.de/
- http://dir.yahoo.com/Business_and_Economy/Business_to_Business/Education/Teaching_and_Learning_Aids/Supplementary_Materials/Software/
- http://dir.yahoo.com/Education/Instructional_Technology/Online_Teaching_and_Learning/Teacher_Resources/
- http://www.thinkquest.org/library/index.shtml
- http://www.tssoftware.com/main.html
- http://www.calculus.net/ci2/?tag=9200438920658

Kollektionen von Lernsoftware für Schule

- http://www.klett-verlag.de/lehrer/mat_nat/medio/
- http://www.klett-verlag.de/heureka/lernsoftware/produkte/produkte.htm
- http://www.westermann.de/multimedia/

Autorensysteme allgemein

- http://www.mcli.dist.maricopa.edu/authoring/
  Suchmaschine für Autorentools.
- http://search.netscape.com/Computers/Systems/Macintosh/Software/Multimedia_Authoring
  Suchmaschine für Autorentools.
- http://solutions.sun.com/catalogs/now/Education_Computer_Based_Training/Authoring_Tools
  Suchmaschine für Autorentools.
- http://distancelearn.about.com/education/distancelearn/msubtools.htm?iam=mt
  Linkliste von Autorentools für Lehrer aus mehreren Bereichen.
- http://www.pierian.com/products/authoring_tools/digital_chisel3/dc3.htm
  Autorensystem zur Erstellung von Webseiten mit Lerninhalten
- http://www.metacard.com/
  CBT-Autorensystem
- http://www.mallardsys.com/lplus/
  Lehrertool für Sprachen

Webbasierte Autorensysteme

- http://demo.cstudies.ubc.ca/
  Liste von Web-Tools.
- http://www.filename.com/wbt/pages/wbttools.htm
  Liste von Web-Tools.
- http://www.onlineeducation.net/welcome.html
  Zum Erstellen von Online-Kursen für die Schule.
- http://www.cict.co.uk/software/textoys/index.htm
  Zum Erstellen von Online-Kursen für die Schule.

Generische Lernsoftware für Naturwissenschaften

- http://www.zum.de/ZUM/public/scheibler.html
  Software für den Physikunterricht.

Generische Simulationen

- http://ecedweb.unomaha.edu/stockmg.htm
  Finanzsimulation
- http://jersey.uoregon.edu/vlab/
  Simulationen für Astrophysik, Mechanik usw.
- http://www.merlan.ca/science/simul.html#frogway
  Für den Biologieunterricht: Sezierung eines Frosches.
- http://math.rice.edu/~joel/NonEuclid/
  Mathematiksimulation.
- http://www.ecobeaker.com/
  Simulationen von Evolutionen.
- http://www.ncsa.uiuc.edu/edu/icm/
  Simulationen von Sonnenaufgängen usw.
- http://www.ercim.org/publication/Ercim_News/enw33/launonen.html
  Simulation über Presse und Druck.
- http://www.shodor.org/master/
  Simulationen für verschiedene Wissenschaften.
- http://modelscience.com/products.html
  Bietet die Möglichkeit Experimente zu simulieren, die ansonsten zu teuer oder zu gefährlich wären.

Editoren für Tests, Quizzes, etc.
Die in dieser Liste aufgezählten Editoren erzeugen Lernsoftware und sind daher meist als generative Tools zu werten.

- http://www.cluesoftware.de/lernsoftware.html
  Programm zur Erstellung von mathematischen Aufgaben für Schüler.
- http://www.netacc.net/~crossdwn/
  Zur Erstellung von Kreuzworträtseln

- http://test.cln.iupui.edu/cpt/help/iq_topic51.htm
  Tool zum Erstellen von Multiple-Choice Tests.
- http://www.4teachers.org/tools/index.shtml
  Software hilft ein Quiz zu erstellen oder selbst annotierte Online Stunden zu entwerfen
- http://www.2learn.ca/teachertools/teachertools.html
  Software, die das Erstellen von Unterrichtsmaterial unterstützt
- http://www.exambuilder.com/
  Allzweck-Prüfungsbogen
- http://www.classbuilder.com/testitems.htm
  Tool zum Erstellen von Klassenarbeiten.

3.3 Beispielsystem I: InterTalk

Das generisch/generative Lehr-/Lernsystem, das wir näher betrachten ist *InterTalk* [18]. Der Grundgedanke des Programms InterTalk 1.0 besteht darin, daß zwei Lernende über ein Netzwerk miteinander verbunden sind und als Team versuchen, gemeinsam Übungsaufgaben zu lösen. Bei der Bearbeitung der Aufgaben haben die Schüler die Möglichkeit, über einen sog. Chatroom schriftlich in der Fremdsprache miteinander zu kommunizieren. InterTalk 1.0 überprüft dabei mit Hilfe eines integrierten Wörterbuches, ob die Lernenden im Chatroom die Fremdsprache benutzen; die Muttersprache ist nicht zugelassen und taucht lediglich als ***, d.h. verschlüsselt und ohne Nutzen für die Schüler auf.

Die Mitspieler müssen zusammenarbeiten, um ein Lösungswort (key) herauszufinden, das sie zur jeweils nächsten Aufgabe bringt. Start und Ziel der Sammlung von Übungsaufgaben (des sog. Plots) sind festgelegt. Die Lernenden können sich den Weg, den sie beschreiten wollen, um an das vorgegebene Ziel zu gelangen, aussuchen. Dies geschieht dadurch, daß sie entweder den sog. *regular key* herausfinden, der sich durch das Lösen der Aufgabe ergibt, oder den *alternative key*, den sie durch Beantworten einer Zusatzfrage finden. Durch die Wahlmöglichkeiten ist der Selbststeuerungsgrad des Programms relativ hoch.

Mit InterTalk 1.0 können neue Lernstoffe mit Übungen präsentiert oder Wiederholungsübungen zu schon bekannten Lerninhalten erstellt werden; dies hängt von den Lernzielen der jeweiligen Lehrperson ab. Die Lehrperson hat die Möglichkeit, während der Bearbeitung alles aufzuzeichnen, was die Schüler eingeben, d.h. ein sog. Logbuch zu erstellen, das für die Auswertung der Übungen und Vorbereitung weiterer Aufgaben nützlich ist. Dieses Logbuch gibt sowohl die Dauer der Bearbeitung der Aufgaben durch die einzelnen Teams als auch die gewählten Lösungswörter an. Außerdem kann sich der Lehrer einschalten, während Schüler Aufgaben bearbeiten. Sowohl einzelne Schüler als auch die gesamte Bearbeitung des Plots durch alle Teams können eingesehen werden. Wird InterTalk 1.0 in einer Klasse im Unterricht eingesetzt, so findet dies in einem sog. Intranet statt, d.h. die Schüler arbeiten offline innerhalb des Computernetzwerks der Schule. Im Idealfall arbeiten Schüler unterschiedlicher Nationalitäten zusammen, indem sie online über das Internet miteinander verbunden sind und in der jeweiligen Fremdsprache die Aufgaben lösen. Ein Beispiel: Arbeitet ein amerikanischer Schüler mit einem deutschen Schüler zeitgleich, d.h. synchron über das Internet zusammen, so muß der Amerikaner Aufgaben auf Deutsch, der Deutsche Aufgaben auf Englisch lösen. Im Chatroom schreiben beide in der jeweiligen Fremdsprache, d.h. der Amerikaner auf Deutsch, der Deutsche auf Englisch. Die Übungssoftware InterTalk 1.0 kann wie oben schon erwähnt nur in einem Netzwerk mit dem InterTalk Server (Host) und mehreren miteinander vernetzten Computern (InterTalk Clients) betrieben werden. Außerdem ist ein Internet-Browser erforderlich, um die Übungssoftware in einer sog. Laufzeitumgebung entweder im Internet oder im Intranet zu laden.

3.3.1 Erstellung von Lerneinheiten durch Lehrer

Die Übungen, nämlich Packages und Plots, werden mit dem Autorenwerkzeug *UPLManager* erstellt. Die Ploterstellung beginnt damit, daß der Lehrer mit Hilfe der verfügbaren UnitBuilder entsprechende Übungen (z.B. Kreuzworträtsel, Lückentext, ...) erstellt. Da die entsprechenden Übungen später im Team von jeweils zwei Schülern durchgeführt werden sollen, werden zwei HTML-Grundgerüst-Dateien generiert. Diese zwei HTML-Grundgerüst-Dateien enthalten neben einem Eintrag für ein UnitBuilder-spezifisches Applet (die Unit soll ja später in einem Webbrowser ablaufen) auch ein kleines JavaScript-Programm. Dieses Programm erfragt zur Laufzeit den Schülernamen. Nach der optionalen Bearbeitung der beiden HTML-Seiten durch den Lehrer müssen die einzelnen Units mit jeweils ihren beiden HTML-Seiten (und deren Inhalten: Bilder, Videos, Sound) zu Packages zusammengebunden werden. Ein Package stellt für das InterTalk ISDK (Instruction Software Development Kit) eine atomare Einheit dar, weil es zum einen eine komplette Lerneinheit darstellt und zum anderen die einzelnen Stationen innerhalb eines Plots repräsentiert. Eine Package-Datei besteht aus mehreren komprimierten Dateien, die u.a. Informationen über die Art der enthaltenen UnitBuilder-Datei, deren Namen, deren Beschreibung und für welche beiden zu lernende Sprachen diese Unit entworfen wurde, enthält.

Damit der Lehrer bereits erstellte Plots oder Packages auch anderen Lehrern zur Verfügung stellen kann (Wiederverwendbarkeit) wurde für InterTalk ein Datenbanksystem entwickelt. Die Anbindung an dieses DBS erfolgt über den UPLManager. Durch Angabe einer Adresse können mit dem UPLManager verschiedene InterTalk Datenbanken im Internet erreicht werden. Jede Datenbank wird von einem Administrator verwaltet, der für die verschiedenen Benutzer Logins und Paßwörter einrichtet. Die Datenbank an sich liegt im XML-Format vor, so kann sie in bereits existierenden Datenbanksystemen (z.B. Oracle, MS ACCESS) wiederverwendet werden.

Der InterTalkServer Plugin-Mechanismus erlaubt es, die Funktionaliät der InterTalk Laufzeitumgebung zu erweitern, indem Drittanbieter eigene InterTalkServer Plugins, die einer vorgegebenen Schnittstelle entsprechen, entwerfen und diese dann in die Laufzeitumgebung einklinken können (Erweiterbarkeit). Beim Hochfahren des InterTalkServer liest dieser sämtliche Plots, Wörterbücher und vorgefertigte Lernerlisten ein und stellt diese dann einem Lehrer zur Auswahl. Der Lehrer startet einen bestimmten Plot, nachdem er diesen ausgewählt, ein passendes Wörterbuch ausgesucht und die Lernerliste zusammengestellt hat. Zur Laufzeit hat der/die Lehrer die Möglichkeit, Statistiken aufzuzeichnen, um diese später auszuwerten. Die Statistiken werden als XML-Datei aufgezeichnet. So können mit entsprechenden Werkzeugen diese Daten ausgewertet bzw. selektiert werden.

3.4 Beispielsystem II: GANIMAM

Als Beispiel für ein generatives Lernsystem betrachten wir GANIMAM, einen webbasierten Generator für interaktive Animationen abstrakter Maschinen [19, 5]. Häufig werden abstrakte Maschinen als plattformunabhängige Zwischenarchitekturen bei der Übersetzung höherer Programmiersprachen verwendet. Die Anweisungen einer abstrakten Maschine sind für die Übersetzung einer bestimmten Quellsprache oder für Quellsprachen desselben Sprachenparadigmas (imperativ, funktional, logisch, objektorientiert) maßgeschneidert.

Im Folgenden beschreiben wir, wie GANIMAM verwendet werden kann und was das System generiert. Abschließend diskutieren wir dann die Vorteile von GANIMAM und der erzeugten interaktiven Animationen sowohl als Entwicklungswerkzeug, als auch als Teil einer Lernsoftware.

3.4.1 Technischer Überblick

GANIMAM kann über die Webseite des GANIMAL-Projekts [20] aufgerufen werden. Der Benutzer (Lehrer oder Lerner) gibt die Spezifikation einer abstrakten Maschine ein, die dann zum Server geschickt wird. Ein CGI-Skript auf dem Server erzeugt Java-Quellcode, der durch einen Java-Compiler in Klassendateien übersetzt wird. In Kombination mit den Klassen des Basispakets von GANIMAM ergeben die Klassendateien ein interaktives Java-Applet.

Das Applet kann über das Internet geladen werden und der Benutzer kann dann Maschinenprogramme eingeben, das Layout der verschiedenen Teile der visualisierten abstrakten Maschine ändern und die animierte Ausführung seines Maschinenprogramms steuern.

Abstrakte Maschinen können eine unterschiedliche Anzahl von Registern, Kellern, Halden haben, daher wird für jede generierte Maschine ein automatisches Layout benötigt. Das automatische Layout gruppiert die verschiedenen Speicherarten um den Akkumulator, eine konzeptionelle Recheneinheit, herum. Programmcode und Keller werden links, die Halde rechts, lokale Variablen über und Register unter dem Akkumulator plaziert. Mit dem Akkumulator ist ein Akkumulatorfenster verbunden, das den Ausdruck anzeigt der gerade im Akkumulator berechnet wird, sowie die Definition der Instruktion bzw. Funktion, die gerade ausgeführt wird. Durch einen zweifachen Mausklick mit der rechten Maustaste auf eine Instruktion im Programmcode wird die Definition der Instruktion in das Akkumulatorfenster geladen. Ein zweifacher Mausklick mit der linken Maustaste auf eine Instruktion im Programmcode setzt den Wert des Programmzählers auf die Adresse der Instruktion, d.h. die Ausführung des Programms wird an dieser Stelle fortgesetzt. Ein Klick auf eine Zelle des Kellers, der Halde oder auf ein Register öffnet ein Fenster. In diesem Fenster kann der Benutzer den Wert und den Typ, bei Registern nur den Wert ändern.

3.4.2 Vorteile interaktiver Animationen

GANIMAM bietet verschiedene Arten der Interaktion. Erstens kann der Benutzer Spezifikationen einer abstrakten Maschine eingeben oder ändern [4]. Beim Visualisieren einer abstrakten Maschine ist es wichtig, daß man nicht nur Daten und Vorgänge auf dem geringen Abstraktionsniveau und der Granularität der Spezifikationssprache darstellt, sondern auch Abstraktionen auf höherem Abstraktionsniveau. Um dies zu ermöglichen, wurden Visualisierungsannotationen zur Spezifikationssprache hinzugefügt. Diese Annotationen ähneln den 'interesting events' einschlägiger Algorithmenanimationssysteme [1].

Nach der Erzeugung der Implementierung der abstrakten Maschine kann der Benutzer Programme in der Sprache der abstrakten Maschine eingeben, diese Schritt für Schritt ausführen und den Inhalt jedes Registers und jeder Speicherzelle inspizieren. Während der Ausführung einer Instruktion zeigt eine Animation den Fluß der Information von Registern oder Speicherzellen in den Akkumulator und vom Akkumulator zurück zu Register oder Speicherzellen. Die Berechnung, die im Akkumulator durchgeführt wird, wird in einem separaten Fenster angezeigt.

Annotationen helfen nur solche Prinzipien sichtbar zu machen, die wir schon vorher kennen. GANIMAM kann auch dazu verwendet werden, neue Prinzipien durch Experimentieren mit Maschinenprogrammen oder Spezifikationen von abstrakten Maschinen zu entdecken. Dieses experimentelle Vorgehen kann u.a. für folgende Zwecke eingesetzt werden:

- Als Teil einer explorativen Lernsoftware ermöglicht es Schülern und Studenten, neue Hypothesen zu formulieren und diese durch Änderung der Spezifikation oder des Maschinenprogramms zu überprüfen. Zusätzliche Hinweise in textueller Form sollte dabei sicherstellen, daß der Lerner auch die wesentlichen Punkte untersucht. Solche könnten z.B. der Vergleich zwischen caller-save-registers und callee-save-registers oder zwischen lazy und eager Auswertung sein oder das Finden des Kellerrahmens des statischen Vorgängers [10]. Andererseits können Lehrer GANIMAM derart instanziieren, daß sie etwa eine abstrakte Maschine für eine imperative und eine logische Prgrammiersprache generieren und dann von den Schülern/Studenten vergleichen lassen.

- Als Entwicklungswerkzeug kann es helfen, Fehler oder Optimierungsmöglichkeiten zu entdecken. Eine solche Optimierung könnte z.B. die Endrekursion betreffen: Durch das Verfolgen der Ausführung von Beispielprogrammen könnte der Benutzer feststellen, daß bestimmte Informationen in einem Kellerrahmen nach rekursiven Aufrufen nicht mehr benötigt werden.

GANIMAM kann auch von Forschern eingesetzt werden, um ihre neusten Implementierungstechniken vorzuführen oder einfach für Rapid Prototyping.

4 Diskussion

Wie erst kürzlich wieder in einer umfangreichen Studie [2] belegt wurde, ersetzt der Computer für viele Lehramtsstudierende vor allem erst einmal die elektrische Schreibmaschine. Offensichtlich ist der Einsatz von Werkzeugen zur Erstellung von Lernsoftware in der Lehrerausbildung nur dann möglich, wenn die Lehrer über Grundkenntnisse im Umgang mit Computern verfügen, die über die reine Textverarbeitung hinausgehen [9]. Darüber hinaus sollten aber diese Werkzeuge lehrergerecht gemacht werden. Bei InterTalk wurde dies dadurch erreicht, daß von Anfang an Lehrer in die Entwicklung des Lernsystems und der Werkzeuge involviert waren. Die meisten Autorensystem bieten für anspruchsvollere Aufgaben Scriptsprachen, in denen komplexere Zusammenhänge programmiert werden können, oder gar Schnittstellen zu sogenanntem Native Code, d.h. Erweiterungen die in irgendeiner Programmiersprache erstellt wurden. Kurz gesagt, existierende Systeme verlangen Programmierkenntnisse für nichttriviale Funktionalitäten. Es stellt sich die Frage, ob und wie die wichtigsten Programmierkonzepte, wie etwa Zustand und Berechnung, Rekursion, Kontrollstrukturen und Abstraktion, in der Lehrerausbildung vermittelt werden sollten. Die Beantwortung dieser Frage ist jedoch nicht Gegenstand dieser Studie.

Man könnte natürlich auch geringere Ansprüche stellen und den Umgang mit Werkzeugen ohne Programmierung vermitteln. Hier ist aber zu bedenken, daß verschiedene Werkzeuge sehr unterschiedliche graphische Metaphern (z.B. Frames oder Flowcharts) verwenden und es daher kein prototypisches Werkzeug gibt, das man in der Lehrerausbildung einsetzen könnte.

Das gleiche Problem stellt sich, wenn wir den Umgang mit Spezifikationssprachen für generische Lehr-/ und Lernsysteme betrachten. Spezifikationssprachen können sehr vielfältig und stark themenabhängig sein, wie man z.B. bei GANIMAM gesehen hat. Die Vorteile solcher Systeme wurden weiter oben bereits diskutiert. Da es bisher aber kaum existierende, generische und generative Systeme gibt, ist es wichtig, daß solche Systeme in solchen Bereichen entwickelt werden, in denen sie mit herkömmlichen Systemen verglichen werden können. Da es kaum generische Werkzeuge gibt, um Lerninhalte aufzubereiten, gibt es insbesondere fast keine, die beim Erstellen von Lernumgebungen die Lernzielkontrolle miterzeugen (Evtl. Multiple-Choice, ...). In der Tat haben wir bei unseren Recherchen keine einziges solches Werkzeug gefunden.

Literatur

[1] M. H. Brown. *Algorithm Animation*. MIT Press, 1987.

[2] D. Baacke, K. U. Hugger, W. Schweins. *Neue Medien im Lehramtsstudium*. Ergebnistelegramm, Bertelsmann Stiftung, Heinz Nixdorf Stiftung, 2000.

[3] D. Boles, M. Schlattmann.*Multimedia-Autorensysteme: Grafisch-interaktive Werkzeuge zur Erstellung multimedialer Anwendungen*. In LOGIN 18(1998) Heft 1, LOGIN-Verlag, 1998.

[4] S. Diehl, T. Kunze, A. Placzek. *GANIMAM: Generation of Interactive Animations of Abstract Maschines*. In Proceedings of "Smalltalk und Java in Industrie und Ausbildung STJA'97" (in German), pp. 185-190, Erfurt (Germany), 1997.

[5] S. Diehl, T. Kunze. *Visualizing Principles of Abstract Machines by Generation*, Future Generation Computer Systems, Vol. 16(7), 2000

[6] S. Diehl, A. Kerren. *Increasing Explorativity by Generation*. To appear in Proceedings of World Conference on Educational Multimedia, Hypermedia and Telecommunications, EDMEDIA-2000, AACE, Montreal, Canada, 2000.

[7] F. Fünfstück, R. Liskowsky, K. Meißner. *Softwarewerkzeuge zur Entwicklung multimedialer Anwendungen. Eine Übersicht*. Informatik Spektrum, Vol. 23(1), pp. 11-25, 2000.

[8] A. Kerren. *Animation of the Semantical Analysis*. In Proceedings of „8. GI-Fachtagung Informatik und Schule INFOS99" (in German), pp. 108-120, Informatik aktuell, Springer, 1999.

[9] R. Marschall. *Ein (vorläufiges) Konzept für die informatische Grundbildung von Lehramtsstudierenden.* In Proceedings of „8. GI-Fachtagung Informatik und Schule INFOS99“ (in German), pp. 130-139, Informatik aktuell, Springer, 1999.

[10] R. Wilhelm, D. Maurer. *Compiler Design: Theory, Construction, Generation.* Addison-Wesley, 1995.

[11] http://www-is.informatik.uni-oldenburg.de/~dibo/teaching/mm98/buch/

[12] http://www.tiac.net/users/jasiglar/faq_index.html

[13] http://www.hyperactivesw.com/

[14] http://www.asymetrix.com/

[15] http://www.macromedia.com/software/director/

[16] http://www.macromedia.com/software/flash/

[17] http://www.macromedia.com/software/authorware/

[18] http://www.cs.uni-sb.de/~diehl/InterTalk/

[19] http://www.cs.uni-sb.de/GANIMAL/GANIMAM/

[20] http://www.cs.uni-sb.de/GANIMAL