teachTool – Ein Autorensystem mit didaktischer Benutzerunterstützung

K. Blankenagel

Zur Erlangung des akademischen Grades eines

Doktors der Naturwissenschaften

vom Fachbereich Mathematik und Naturwissenschaften der
Bergischen Universität Wuppertal
genehmigte

Dissertation

von

Dipl.-Math. Karsten Blankenagel

aus Wuppertal

Tag der mündlichen Prüfung:	19.10.2006
Erstgutachter:	Prof. Dr. A. Frommer
Zweitgutachter:	Prof. Dr. M. Stein

Inhaltsverzeichnis

Kapitel 1

Einleitung

Bei der Erstellung von E-Learning-Seiten für das Internet, stellen sich stets vier zentrale Fragen:

- Wie erzeugt man möglichst komfortabel Interaktionen?
- Welchen strukturellen Aufbau sollen die Seiten besitzen?
- Welcher didaktische Aufbau ist zu wählen?
- Welches Design ist passend?

Eine Grundanforderung an ein Autorensystem ist, dass (web-basierte) Lerneinheiten (im Weiteren auch „Module“ genannt) ohne Programmierkenntnisse schnell und einfach erstellt werden können.
Darüber hinaus sollte ein solches Programm auf die oben gestellten Fragen leicht nachvollziehbare Antworten liefern und den Autor bei der entsprechenden Realisierung unterstützen.

In dieser Arbeit wird der Entwicklungsprozess des Autorensystems teachTool vorgestellt. Die während dessen gemachten Erfahrungen zeigen, dass es bei der Entwicklung eines Autorensystems förderlich ist, sich an bestehenden E-Learning-Projekten zu orientieren, die auf die obigen Fragen eindeutige und erprobte Antworten bereitstellen. Durch dieses Vorgehen widersteht man der Versuchung, sämtliche Angebote des Mediums HTML durch das Autorenwerkzeug zu ermöglichen. Was sich hier als Einschränkung der erzeugbaren Lerneinheiten anhört, stellt für deren Autoren eine wertvolle Hilfe dar, bei der Bestrebung die Lernerfreundlichkeit ihrer Module zu optimieren.

Das E-Learning-Projekt MathePrisma[1] [Arb] der Bergischen Universität Wuppertal hat sich in seiner Konzeptionierungsphase mit den obigen vier Fragen sehr ausführlich befasst. MathePrisma ist eine inzwischen breit genutzte Modulsammlung zu

[1] http://www.matheprisma.de/ (Alle in dieser Arbeit angegebenen Internetadressen waren gültig am 04.07.2006.)

Themen der Mathematik und Informatik. Das Projekt wurde im Jahr 2001 von der Gesellschaft für Medien in der Wissenschaft e.V. mit dem internationalen mediendidaktischen Hochschulpreis MEDIDA-PRIX ausgezeichnet (siehe [FKV04]).
Die Attraktivität von MathePrisma beruht ganz wesentlich darauf, dass als Antwort auf die obigen Fragen verbindliche Regeln entwickelt wurden. Auf diese Weise gelang es, Übersichtlichkeit in der Darstellung, spezifische Nutzung des Mediums durch Interaktionen und einen (für das Projekt) sinnvollen didaktischen Aufbau zu schaffen und erhaltbar zu machen.
Weitere Informationen zur Konzeption, Entwicklung und Erprobung des Projekts MathePrisma finden sich in [Kri03] und [FGK00].

Zum Autorenkreis der Module gehören neben den Kernentwicklern an der Universität Wuppertal in zunehmendem Maße weitere Hochschulangehörige, Lehrer an weiterführenden Schulen, Studierende und Schüler.
Hieraus erwuchs die Forderung, ein Autorenwerkzeug zu erstellten, welches das Schreiben von Modulen auch ohne Kenntnisse in HTML und JavaScript möglich macht und darüber hinaus zusätzlich das Einhalten der MathePrisma-spezifischen Designregeln und didaktischen Vorgaben unterstützt und sogar verifiziert (siehe [BKV04]).
Das in dieser Arbeit vorgestellte Programm teachTool erfüllt diese Anforderungen und ermöglicht zusätzlich, die Designvorgaben auf einen benutzerdefinierten Standard anzupassen. Ferner können die zu überprüfenden didaktischen Regeln individuell vordefiniert werden.

Die mit teachTool erzeugten Module werden als LaTeX-Dateien abgespeichert. Diese wurden in älteren Versionen[2] des Programms mit Hilfe von latex2html und den in dem Paket Latex2mp[3] festgelegten Makros nach HTML übersetzt. LaTeX stellt einen Standard beim Verfassen mathematischer Texte dar.
In der aktuellen teachTool Version wird auf den Einsatz von latex2html Verzichtet, da latex2html erstens sehr lange Zeit für die Übersetzung nach HTML benötigt und zweitens die Installation von latex2html unter Windows sehr aufwändig ist. teachTool generiert jetzt die HTML-Seiten der Lerneinheiten intern unter Verwendung der in Latex2mp definierten Makros.
Durch das Abspeichern der Module im LaTeX-Format sind diese unabhängig von ihrer Präsentationsform. Sollte es in Zukunft nötig sein, das Design der Internetseiten zu ändern, müssen lediglich die Makros abgeändert und die Module neu übersetzt werden. Ebenso verhält es sich bei Veränderungen der Browser-Standards, die die Befehle für JavaScript festlegen.
Sollte aufgrund neuer lerntheoretischer Erkenntnisse eine inhaltliche Umstrukturie-

[2] Der in dieser Arbeit beschriebene Ist-Zustands von teachTool bezieht sich auf die Version 1.0, die zum Zeitpunkt der Fertigstellung der Arbeit existierte.

[3] Latex2mp wurde von PD M. Peter (Universität Freiburg) erstellt und zusammen mit dem Autor weiterentwickelt.

rung erforderlich sein, können die Module, wie in Kapitel 4 beschrieben, mit teachTool durch Drag&Drop neu aufgebaut werden.
Eine Printversion kann auf eine ähnliche Weise erzeugt werden, wie die HTML-Version. Hierfür werden die LaTeX-Dateien der Module mit pdflatex verarbeitet.

Die am Anfang gestellten Fragen bestimmen durch ihre grundlegende Bedeutung den Aufbau dieser Arbeit in den ersten Kapiteln.
Nach einer kurzen Definition der Begriffe „Autorensysteme“ und “Lerneinheiten“, wie sie in dieser Arbeit verwendet werden, werden in Kapitel 3 die technischen Grundelemente von E-Learning-Einheiten zusammengefasst. Diese Grundelemente können in drei Gruppen eingeteilt werden:

- strukturelle (Kapitel, Seiten, ...)
- stilistische (Bilder, Textformat, ...)
- interaktive (Aufgabentypen, Links, ...)

Besondere Aufmerksamkeit erhalten dabei die interaktiven Elemente. Hier wird also auf die Frage „Wie erzeugt man möglichst komfortabel Interaktionen?“ detailliert eingegangen.
Das Kapitel zerfällt in die Bereiche „Konzept bei MathePrisma“, „Umsetzung in teachTool “ und „Vergleich mit anderen Autorensystemen“. Dieser Aufbau wird auch in den folgenden drei Kapiteln angewendet.
Es wird also jeweils angegeben, welche Elemente und Regeln sich durch die Erfahrungen im Projekt MathePrisma als für E-Learning förderlich bzw. hinderlich herausgestellt haben. Danach wird beschrieben, wie diese Überlegungen in teachTool berücksichtigt wurden und die Stärken und Schwächen von teachTool mit denen anderer Autorensysteme verglichen.
Gerade im Bereich der technischen Aspekte von Autorensystemen ist wenig Literatur vorhanden. „Das Thema 'Entwicklungswerkzeuge für E-Learning' wurde bis vor kurzem weitgehend in der E-Learning-Diskussion (Literatur, Messen, Kongresse, Studien) vernachlässigt, es stand einerseits das Thema E-Learning allgemein mit den damit erwarteten Vorteilen im Vordergrund, andererseits das Thema Lernplattformen (LMS), wobei hier vor allem technische Aspekte dominierten.“ (siehe [Fre03])
Daher wurden zum Einordnen des Funktionsumfangs von teachTool in erster Linie andere Autorensysteme zum Vergleich herangezogen.
In Kapitel 4 werden die Strukturierungsmuster für Web-Seiten

- Sequenz
- Gitternetz
- Hierarchie
- Spinnennetz

betrachtet und beschrieben, wie diese in MathePrisma, teachTool und anderen Autorensystemen Berücksichtigung finden.
Das folgende Kapitel setzt sich mit der Frage „Welcher didaktische Aufbau ist zu wählen?“ auseinander. Das didaktische Konzept von MathePrisma ist problem- und handlungsorientiert und „gemäßigt“ konstruktivistisch. Es hat sich in einem langen Evaluationsprozess, der in [Kri03] geschildert wird, sehr bewährt und wird deshalb in dieser Arbeit nicht mit behavioristischen oder kognitivistischen Modellen verglichen, sondern nur grundlegend beschrieben. Es wurden in teachTool Mechanismen implementiert, die dem E-Learning-Autor die Festlegung und Einhaltung didaktischer Regeln erleichtern. Ferner werden in diesem Kapitel die Einsatzmöglichkeiten von teachTool in der Lehrerausbildung dargestellt.
Im Weiteren geht es um die Wahl eines Designs für E-Learning-Einheiten und die Unterstützung, die teachTool bei der Generierung (mathematischer) Formeln bietet.
Kapitel 8 und 10 beschäftigen sich mit dem Thema Nachhaltigkeit. Im ersteren wird der Standard SCORM beschrieben und wie durch diesen Lerneinheiten, die mit teachTool erstellt werden, auch in Zukunft und auf verschiedenen Lernplattformen, verwendet werden können. Kapitel 10 erläutert das Software-Design von teachTool. Es wird hier insbesondere darauf eingegangen, wie weitere Grundelemente (z.B. weitere Interaktionsformen) implementiert werden können und wie die HTML-Ausgabe der Lerneinheiten für zukünftige Entwicklungen angepasst werden kann. Das Kapitel richtet sich also in erster Linie an Java-erfahrene Programmierer.
Das von diesen beiden eingeschlossene Kapitel 9 zeigt auf, wo bei der Implementierung von teachTool Überlegungen aus der Software-Ergonomie verwendet wurden, um die Benutzerfreundlichkeit des Programms zu gewährleisten.
Das vorletzte Kapitel „Einsatz und Evaluation“ gibt die Resultate der Befragungen von drei Anwendergruppen wieder.
Die bei den Betrachtungen in dieser Arbeit hervorgetreten Vor- und Nachteile von teachTool werden abschließend in der Zusammenfassung noch einmal aufgeführt.

Kapitel 2

Autorensysteme und Lerneinheiten

In diesem Kapitel werden die Begriffe Autorensysteme und die mit ihnen erzeugbaren Lerneinheiten definiert. Außerdem wird eine Liste häufig verwendeter Autorensysteme und Gründe für die Implementierung von teachTool angegeben.

2.1 Funktionen und Eigenschaften von Autorensystemen

Autorensysteme „entstanden als Entwicklungswerkzeuge für professionelle Anwendungen (Applikationen) aus Programmiersprachen mit dem Ziel, auch solchen Personen das Schreiben von Lehr- und Lernprogrammen (zu) ermöglichen, die kaum über Programmierkenntnisse verfügen. Der Programmautor kann sich damit im wesentlichen auf die inhaltliche und didaktisch-methodische Gestaltung des Programms konzentrieren. Den eigentlichen Programmcode erzeugt das Autorensystem.“ (siehe [Sac90], Seite 117)

Autorensysteme lassen sich in drei Bereiche klassifizieren, die besonders die Arbeitsweise des Programms berücksichtigen (siehe [Hey00]).

- seitenorientiert
  Ähnlich einem Buch werden hier Lerneinheiten erstellt, indem Kapitel und Seiten angelegt werden. Durch die Verwendung von Links kann dem Lernenden zusätzlich ermöglicht werden die Reihenfolge der angezeigten Seiten individuell zu bearbeiten. Vertreter dieser Kategorie sind ToolBook [Sum] und IDEA [Lin].

- iconorientiert
  Bei diesen Programmen werden Grundelemente wie Texte, Bilder und Interaktionen durch Icons dargestellt, die in einem Flussdiagramm angeordnet werden. Als Beispiel für diese Klasse fungiert das Programm Authorware [Adoa].

- zeitachsenorientiert
  Mit diesen Autorensystemen können in erster Linie Lerneinheiten in Form von Präsentationen oder Animationen erstellt werden. Typische Vertreter sind Director [Adob] und Flash [Adoc].

Bei einigen Programmen kann eine Zuordnung in eine Klasse nicht eindeutig vorgenommen werden. So arbeiten z.B. seitenorientierte Autorensysteme gerade bei der Darstellung der Struktur einer Lerneinheit zum Teil iconorientiert. Zu diesen Programmen gehört auch teachTool.

Wie lassen sich die Lerneinheiten charakterisieren, deren Erstellung in dieser Arbeit untersucht wird?
Mit dieser Frage beschäftigte sich auch M. Peter, der das Tool Latex2mp grundlegend entwickelt hat, mit dem man aus LaTeX-Dateien MathePrisma-Module erzeugen kann. Latex2mp stellt eine wichtige Grundlage für teachTool dar. Daher besitzen Lerneinheiten, die mit Latex2mp erzeugt werden können, in großen Teilen Charakteristika, die für diese Arbeit von Bedeutung sind. M. Peter beschreibt diese Lerneinheiten in [Pet04] als „interactive multimedia hypertexts (IMMHs) with mathematical content“. Eine genauere Beschreibung der einzelnen Begriffe kann in Anlehnung an M. Peter wie folgt gegeben werden:

1. Multimediale Aspekte finden sich schon in gedruckten Dokumenten in statischer Form durch den Einsatz von Bildern und Graphiken. In elektronischen Dokumenten gibt es zusätzlich noch dynamische Elemente wie Filme und Ton.

2. Interaktive Dokumente besitzen nicht nur die genannten statischen und dynamischen multimedialen Elemente, sondern sie können auch auf komplexe Art und Weise auf Eingaben des Benutzers reagieren und so mit diesem in Interaktion treten. Dies geschieht z.B. über verschiedene Aufgabenformate oder über Schritt-für-Schritt-Beschreibungen wie Diashows (siehe Kapitel „Grundelemente“).

3. Hypertexte geben durch den Einsatz von Hyperlinks die Möglichkeit, Seiten nicht nur in der vom Autor vorgegebenen Reihenfolge linear hintereinander zu betrachten, sondern auch eigene (Lern-) Pfade zu verfolgen. Dadurch erscheinen die Dokumente zum einen viel lebhafter. Zum anderen kann damit auch eine höhere semantische Ebene erreicht werden.

Die wichtigsten Funktionen und Eigenschaften von Autorensystemen, die in diesem Abschnitt beschrieben werden, sind in der Liste auf der folgenden Seite aufgeführt.

- Erzeugung von Lerneinheiten im HTML-Format
- Erstellung von Offline- bzw. CD-Versionen der Module
- WYSIWYG
- Plattformunabhängigkeit
- Einsetzbarkeit der Lerneinheiten auf verschiedenen Browsern
- Didaktische Unterstützung der Autoren
- Komfortable Generierung einiger Interaktionsformen
- Eingabe mathematischer Formeln
- Berücksichtigung von Standards (z.B. SCORM)
- Erweiterbarkeit (siehe Kapitel 10)

Das Internet biete eine geeignete Plattform, auf der die erstellten Lehr- und Lernprogramme verbreitet werden können. Daher generieren neuere Autorensysteme Lerneinheiten in einem Format, das für das Internet verwendet werden kann (HTML). Trotzdem sollte auch eine Offline-Version als Ausgabe vom Autorensystem angeboten werden, da es bei einem Einsatz in Unterrichtssituationen vorteilhaft ist, sich nicht auf die Verfügbarkeit des Online-Angebots verlassen zu müssen. Ferner ist es von Vorteil, wenn das Angebot auch über eine CD verbreitet werden kann.

Autorensysteme haben sich aus Autorensprachen entwickelt, um deren Nutzbarkeit für einen größeren Autorenkreis zu ermöglichen. Es ist daher nur konsequent, dass sie nach dem Prinzip WYSIWYG („What You See Is What You Get“) funktionieren und womöglich Standardbedienweisen besitzen, die von anderen Softwareprodukten bekannt sind (siehe Kapitel „Software-Ergonomie“). Dies stellt ein Problem dar bei HTML-basierten Autorensystemen. Sollen z.B. formatierte Texte über einen Browser eingegeben werden, werden in den im Browser angezeigten HTML-Seiten Eingabefelder (textarea) verwendet. Diese unterliegen einigen technischen Einschränkungen. So kann man z.B. nicht, aufgrund eines Knopfdrucks den Textstil ändern oder Bilder anzeigen lassen. Deshalb werden oftmals Techniken eingesetzt, die im Text vordefinierte Kennzeichnungen (tags) anzeigen. HTML-basierte Autorensysteme finden wegen dieser Einschränkungen in der vorliegenden Arbeit keine Berücksichtigung.

Ein weiteres technisches Kriterium, das Autorensysteme erfüllen sollten, ist die Plattformunabhängigkeit. In erster Linie dominieren gerade auf Computern, die von Lehrern genutzt werden, Windows Betriebssysteme. Trotzdem ist der Anteil an Computern, auf denen Linux oder Mac OS eingesetzt wird, nicht zu vernachlässigen. Außerdem werden neben dem am meisten genutzten Browser Internet Explorer auch

häufig Netscape oder Opera verwendet. Autorensysteme sollten daher möglichst unter allen genannten Betriebssystemen eingesetzt werden können. Nur dann ist z.B. ein gemeinsames Bearbeiten einer Lerneinheit sichergestellt, wenn von den Bearbeitern verschiedene Betriebssysteme verwendet werden. Eine günstige Basis für die Implementierung von Autorensystemen ist die Programmiersprache Java. In Java geschriebene Programme benötigen lediglich ein Java Runtime Environment, das für fast alle Betriebssysteme vorhanden ist.

Außerdem sollten die erstellten Lerneinheiten auf allen gängigen Browsern zum Lernen eingesetzt werden können. Die Rechnerausstattung an Schulen entspricht zum Teil nicht den neusten Entwicklungsstandards. Die Internetseiten der Lerneinheiten dürfen daher nicht ausschließlich auf modernsten Standards in HTML oder JavaScript aufbauen. Es ist aus Sicht des Autors auch nicht sinnvoll, wenn Lerneinheiten dazu auffordern einen neuen Browser zu installieren, nur um das Aussehen der Seiten geringfügig zu verbessern. Was das Design und die Bedienbarkeit von Internetseiten angeht, ist schon in älteren Standards (fast) alles realisierbar.

Im Zitat zu Beginn dieses Kapitels wurde darauf hingewiesen, dass ein Autor durch die Nutzung eines Autorensystems in der Lage sein sollte, sich ausschließlich auf die inhaltliche und didaktisch-methodische Gestaltung seiner Lerneinheiten zu konzentrieren. Eine Forderung an moderne Autorensysteme ist deshalb die Bereitstellung von Funktionen, die den Autor auch in diesen Belangen unterstützen. Gerade diese Möglichkeit ist jedoch bei vielen Autorensystemen wenn überhaupt nur sehr eingeschränkt vorhanden (siehe Kapitel 5 „Didaktik“).

Eine weitere Funktion, die Autorensysteme besitzen sollten, ist die Möglichkeit der Eingabe mathematischer Formeln. Diese wird nicht nur für die Erstellung von Lerneinheiten für die Mathematik oder Naturwissenschaften benötigt. Sie findet Verwendung in allen Gebieten, die mathematische Sachverhalte betrachten wie z.B. Verkehrstechnik oder Maschinenbau.

Gerade im Umfeld von E-Learning wird die Frage nach der Nachhaltigkeit und Wiederverwendbarkeit von Lernmaterialien groß geschrieben. Es haben sich daher Standards entwickelt, die sicherstellen (sollen), dass entwickelte Lerneinheiten auch nach zukünftigen technischen Weiterentwicklungen verwendet werden können. Allem voran sei an dieser Stelle der Standard SCORM (Sharable Content Object Reference Model) genannt. Obwohl SCORM im Bereich der pädagogischen Metadaten noch verbesserungsfähig ist, sollten Autorensysteme ihn bereits heute unterstützen.

Die aufgeführten Funktionen und Eigenschaften von Autorensystemen decken sich zu großen Teilen mit den Anforderungen, die in [CH02] als Checkliste zur Auswahl von Autorensystemen zusammengestellt wurden. Sie gliedern sich in [CH02] in die Bereiche technologische, betriebliche und pädagogische Anforderungen.

2.2 Häufig verwendete Autorensysteme

Die Anzahl der angebotenen Autorensysteme ist sehr groß. Bei seinen Recherchen ist der Autor auf 160 Produkte gestoßen, die als Autorensysteme genannt wurden. Bei einigen dieser Programme ist jedoch anzunehmen, dass sie zur Gruppe der Learning Management Systeme (LMS) gehören oder nur in Verbindung mit einem LMS bedient werden können (z.B. IBT Tools). Andere Programme eignen sich mehr zur Erstellung von Präsentation als zur Erzeugung von E-Learning-Einheiten (z.B. ViewletBuilder).
Um einen Überblick über die häufig verwendeten Autorensysteme zu bekommen, wurden 7 Produktvergleiche für Autorensysteme aus den Jahren 2001 bis 2005 als Grundlage genommen und ermittelt, wie oft die jeweiligen Programme in den Vergleichen vorkamen.
Es folgt die Liste der berücksichtigten Studien. Die ersten 6 Studien werden in [Fre03] zusammengefasst:

- Authoring Systems Buyers Guide (Hall 2001) [Hal]
- Learning Management Systems & Authoring Tools (Projekt IST 2001) [Inf01]
- Werkzeuge für WBT&Co. (managerSeminare, e-le@rning 2001) [Wal01]
- Autorenwerkzeuge für Online-Lernangebote (Kerkau 2002) [Ker02]
- E-Learning-Produkte im Vergleich (DLC-Studie, Personalwirtschaft 2002) [Chi02]
- Jeder kann jetzt Autor sein (Wirtschaft & Weiterbildung 2002) [Pay02]
- Authoring Tool KnowledgeBase 2006 (Hall 2005) [NV05]

Dieses Vorgehen ermöglichte es, eine Rangfolge der Autorensysteme aufgrund der Anzahl der Nennungen in den berücksichtigten Studien zu erstellen. Zusätzlich wurde der Bekanntheitsgrad der Programme ermittelt, indem die Anzahl der von Google gefundenen Internetseiten mit Produkt- und Anbieternamen bestimmte wurde. Die Autorensysteme mit mindestens drei Nennungen bei den ausgewählten Studien sind sortiert nach der Anzahl der Nennungen und den Google Treffern in Tabelle 2.1 aufgelistet.

Die bekanntesten Autorensysteme sind demnach ToolBook und Authorware. Das wird auch von Freibichler in [Fre03] bestätigt, der die Programme von Macromedia und ToolBook als marktführend bezeichnet. ToolBook Instructor 2004 (Version 8.90.95) wurde daher auch zum Vergleich mit teachTool untersucht. Zusätzlich wurde ein Autorensystem ausgewählt, bei dem wie in teachTool die zu bearbeitende

Tool	**Anbieter**	**Studien**	**Google Suche**	**Google Treffer 02.12.2005**
ToolBook	SumTotal	5	ToolBook SumTotal	85.100
Authorware	Macromedia	4	Authorware Macromedia	872.000
IBT Tools	time4you	4	IBT time4you	888
Flash	Macromedia	3	Flash Macromedia	69.500.000
Director	Macromedia	3	Director Macromedia	12.300.000
Hot Potatoes	Language Centre University Victoria	3	“Hot Potatoes“ Victoria	54.700
Dreamweaver mit Coursebuilder	Macromedia	3	Coursebuilder Macromedia	49.500
ViewletBuilder	Qarbon.com Inc.	3	ViewletBuilder Qarbon	31.000
Lectora Publisher	Trivantis Corporation	3	Lectora Trivantis	9.960
Trainersoft	Omniplex	3	Trainersoft Omniplex	1.980
IDEA	Link & Link	3	IDEA “Link&Link“ OR “Link & Link“ OR “Link und Link“	1.490
TutorialMaker	LernQuotient	3	TutorialMaker	1.370
Lecturnity	imc	3	lecturnity im-c OR imc	716
Dynamic Power Trainer	Dynamic Media	3	“Dynamic PowerTrainer“ “Dynamic Media“	666
Course Factory Web	ets	3	CourseFactoryWeb ets	109

Tabelle 2.1: Liste häufig verwendeter Autorensysteme

Lerneinheit in Form eines Strukturbaums dargestellt wird. Das nach Tabelle 2.1 verbreitetste System, das diese Eigenschaft erfüllt, ist Lectora. Dieses lag in Version 2005 SP3a (2538) vor.

2.3 Gründe für die Implementierung von teachTool

Das E-Learning-Projekt MathePrisma der Bergischen Universität Wuppertal ist seit einigen Jahren ein integraler Bestandteil der Lehramtsausbildung. Im Rahmen ihres 1. Staatsexamens haben bisher 12 Studierende Module entwickelt, die sie in ihrer schriftlichen Ausarbeitung beschrieben haben. Des Weiteren kommt dem Bereich Medien- und Vermittlungskompetenz nicht nur in der Lehramtsausbildung, sondern auch in den Curricula anderer Studiengänge eine wachsende Bedeutung zu. Es geht jedoch hierbei nicht darum, den Studierenden, ausführliche Kenntnisse in HTML zu vermitteln, sondern vielmehr, ihnen Programme vorzustellen, mit denen sie in ihrer späteren Tätigkeit an Schulen für ihre eigenen Zwecke E-Learning-Einheiten erstellen können. Zusammen mit der Frage „Mit welchen Programmen kann ich Module entwickeln?“, kommt dann natürlich auch die Frage auf „Wie können die Inhalte didaktisch geeignet für das Medium Computer aufbereitet werden?“.
teachTool ermöglicht die Erstellung aller struktureller und interaktiver Elemente, die sich im Projekt MathePrisma als für E-Learning bedeutsam herausgestellt haben, Programmierkenntnisse werden nicht benötigt. Ausgenommen sind hier selbstverständlich Applets und Flash-Interaktionen, die auch mit keinem anderen Autorensystem erzeugt werden können. Darüber hinaus gibt teachTool den Autoren von Modulen Hinweise, welche didaktischen Grundsätze befolgt werden sollten. Diese Hinweise und die Tatsache, dass Abschnitte mit pädagogischen Metadaten versehen werden können, unterstützen speziell den in der Lehramtsausbildung wichtigen Aspekt des Sequenzierens von Inhalten. Sie sind aber auch allgemein gerade für weniger erfahrene Autoren hilfreich. Eine solche Funktionalität in dem in teachTool implementierten Umfang wird von keinem der während der Recherchen für diese Arbeit dem Autor bekannt gewordenen Autorensysteme zur Verfügung gestellt.
Diese Vorteile für die Lehramtsausbildung sind in keiner Weise auf Wuppertal oder die Mathematik eingeschränkt. So wurde teachTool beispielsweise auch an der Universität Mainz in einem Seminar zur Modellierung und Visualisierung eingesetzt.

Nach Schulmeister [Sch02] haben Rode und Poirot [RP89] in einer Befragung nachgewiesen, dass „Lehrer die angebotenen Autorensysteme nicht nutzen.“ Als für Lehrer wichtigstes Kriterium eines Autorensystems stellte sich „ease of editing“ heraus, also die Möglichkeit mit wenig Aufwand die für den Unterrichtseinsatz benötigten Materialien zu erstellen.
Einen Vorteil des Einsatzes von Autorensystemen in Schulen nennt Heyder in [Hey00] auf Seite 61: „Eine Lösungsmöglichkeit möglichst viele Vorteile in der Computerleh-

re zu vereinigen, bieten Autorensysteme. So kann der Autor (= schülergesteuerter Unterricht) den Weg bestimmen und nicht der Computer, Schüler sind eigenständig tätig, sie können handelnd an das Unterrichtsthema herangehen und damit den Wissenstransfer fördern.“
Gerade die Autorengruppen der Schüler und der Lehrer setzten eine sehr gute Bedienbarkeit und schnelle Erlernbarkeit des Autorensystems voraus. Diese Merkmale sind daher wichtige Aspekte bei der Entwicklung von teachTool gewesen.

E-Learning-Einheiten sollten nicht nur auf den neusten Browsern eingesetzt werden können. Es ist auch mit älteren Browsern sehr gut möglich, Lerneinheiten zu bearbeiten, ohne auf die Vorteile von E-Learning zu verzichten.
Auf den Internetseiten zu ToolBook z.B. gibt es „Showcases“, die mit Netscape 7.1 nicht angezeigt werden können. Die Fehlermeldung besagt:

Currently, only these browsers are supported:

- Microsoft IE 5.5 SP1 and above
- Netscape 7.2 and above
- Firefox and other Mozilla-based browsers

Es ist nicht nur zeitaufwändig, sich stets den aktuellsten Browser zu installieren. Es ist auch nicht einzusehen, warum ältere Browser nicht unterstützt werden.
Ziel von teachTool ist es daher, Lerneinheiten zu erzeugen, die im Wesentlichen auf sämtlichen Browsern auf allen Betriebssystemen funktionieren.
Gleiches gilt auch für das Programm teachTool selbst. Als Programmiersprache wurde Java gewählt, um die Erstellung von Modulen auf allen Plattformen zu unterstützen. Benötigt wird eine Java Virtual Machine ab Version 1.4.2. Der Einsatz des Programms ist unter Windows, Linux und Mac OS X erprobt und sichergestellt worden.

Viele Bildungseinrichtungen wie Schulen und Universitäten haben bei der heutigen Finanzlage Schwierigkeiten, die Kosten für oftmals sehr teure Autorensysteme aufzubringen[1]. Dennoch ist ein Einsatz dieser Programme gerade in Bildungseinrichtungen von großer Bedeutung. teachTool kann dieses Problem lösen, weil es in Zukunft kostenlos zum Download über die MathePrisma-Plattform angeboten wird.

Insbesondere bei der Entwicklung von E-Learning-Angeboten für die Industrie spielt der Begriff des Rapid E-Learning [Wei03] eine immer größere Rolle. Es geht darum Lerneinheiten möglichst schnell und damit kostenminimiert herzustellen und so die

[1] Auf den Internetseiten von SumTotal Systems, Inc. waren am 29.05.2006 folgende Preise für ToolBook Assistant zu finden: Retail Price $1,495, Education Price $1,110, Student Price $149 (http://www.toolbook.com/education/specialoffers.php).

Effizienz des Lernens am Computer zu steigern. Diese Überlegung steht im Einklang mit den Beweggründen für einen Einsatz von E-Learning in der industriellen Fortbildung, nämlich der Kostenersparnis durch den Verzicht auf Lehrpersonal und eventuell anfallende Reisekosten.
Ein Hauptproblem beim Steigern der Produktivität von E-Learning ist das Erhalten bzw. Verbessern der didaktischen Qualität. „Es häufen sich zunehmend kritische Stimmen (z.B. Schulmeister), die den meisten angebotenen E-Learning-Kursen eine 'schlechte' Didaktik bescheinigen, die auch die Akzeptanz von E-Learning auf Nutzerseite beeinträchtige. Welche Rolle spielen dabei Entwicklungswerkzeuge? Eine sehr große, bisher weitgehend vernachlässigte Rolle: Die Entwicklungswerkzeuge legen in beträchtlichem Umfang fest, wie ein E-Learning-Kurs didaktisch-methodisch gestaltet werden kann.“ (siehe [Fre03]) Gerade das Berücksichtigen didaktischer Überlegungen benötigt bei der Planung und Umsetzung von E-Learning-Einheiten sehr viel Zeit und Erfahrung. Es ist daher naheliegend die Verantwortung für den Erhalt der didaktischen Qualität zumindest zum Teil durch Autorensysteme abzudecken. „In der Praxis erweisen sich die gängigen Werkzeuge jedoch in vielfacher Hinsicht als wenig zufriedenstellend. Dies hat verschiedene Forschungsbemühungen motiviert. Eine Richtung versucht, den didaktischen Entscheidungsprozess in Software abzubilden, um diese Entscheidungen selbst 'automatisieren' und damit dem Computer überlassen zu können ('strong automation'). Eine andere, eher zurückhaltende Richtung beschränkt sich auf Computerwerkzeuge, die das didaktische Design zu unterstützen versuchen ('weak automation').“ (siehe [Ker01], Seite 364)
Autorensysteme, die eine sehr starke Automatisierung vornehmen, gestalten in erster Linie den Ablauf der Lerneinheiten aufgrund von Informationen über das Vorwissen und die Lernziele der gewählten Zielgruppe. Bei diesem Vorgehen gibt es zwei Schwierigkeiten, zu denen bisher kaum befriedigende Lösungen existieren. Zum einen müssen Lernobjekte erstellt und mit Metadaten versehen werden. Es ist jedoch auch für erfahrene Autoren nicht leicht, Lernobjekte zu schaffen, die in verschiedenen Kontexten einsetzbar und zugleich motivierend sind. Außerdem besitzt selbst der im E-Learning momentan am weitesten entwickelte Standard SCORM (siehe Kapitel 8) im Bereich der pädagogischen Metadaten einige Schwächen [Paw04]. Zum anderen werden für die automatisierte Sequenzierung Regeln und Mechanismen benötigt, die sehr eng mit dem umfangreichen Forschungsgebiet der künstlichen Intelligenz verwoben sind. Auch hier ist es aufgrund der vielen ungelösten Probleme im Bereich der künstlichen Intelligenz und der großen Anzahl an benötigten Regeln schwierig, befriedigende Automatisierungslösungen zu finden.
Es wäre somit nach Kerres (siehe [Ker01], Seite 376) „ein *unterstützendes* Ratgebersystem ein besserer Weg zu einer Qualitätssteigerung, auch wenn sich dieser Effekt nur indirekt, d.h. über die gesteigerte Kompetenz bzw. Performanz des Entwicklers einstellt.“
Ziel von teachTool ist es, sowohl bei der Steigerung der Produktivität der Erstel-

lung von computergestütztem Lernen im Sinne von Rapid E-Learning als auch im Erzeugen von Lerneinheiten mit hoher didaktischer Qualität die Autoren optimal zu unterstützen. Die Produktivitätssteigerung geschieht in erster Linie über eine hohe Benutzerfreundlichkeit, die ein schnelles Arbeiten fördert. Der Autor wird in didaktischer Hinsicht unterstützt

- durch die Bereitstellung vordefinierter Abschnittstypen (z.B. Aufgabe, Experiment), welche die Strukturierung der Lerneinheiten vereinfachen und
- durch den in teachTool implementierten „Didactics Check“ (siehe Kapitel 5 „Didaktik“), der einem Autor Probleme aufweist und konkrete Hinweise zur Verbesserung der didaktischen Qualität liefert.

Kapitel 3

Technische Grundelemente

Die Grundelemente, aus denen E-Learning-Einheiten bestehen, und die somit von Autorensystemen angeboten werden, können in drei Gruppen aufgeteilt werden:

- Struktur, z.B. Kapitel und Seiten
- (Präsentations-)Stil, z.B. Fettdruck und Bilder
- Interaktion, z.B. Multiple Choice und Drag&Drop-Aufgaben

Interaktionen bilden dabei den wichtigsten Faktor (siehe [Sch03], Seite 156).
Sie bieten die Möglichkeit explorativer Problemlöseprozesse, der Selbstkontrolle des Lernenden, unbegrenzter Trainingsszenarien und stellen ein Alleinstellungsmerkmal von E-Learning gegenüber anderen Lernmedien wie z.B. Büchern dar. Ihnen wird daher in diesem Kapitel besondere Beachtung geschenkt.

3.1 Konzept bei MathePrisma

Die folgenden Grundelemente werden in MathePrisma verwendet:

Struktur
Kapitel, Seite
Nebenpfadseite
Arbeitsblatt
Abschnitte:
    Normaler Text
    Interaktive Abschnitte: Experiment, Aufgabe, Diashow, Film
    Definition/Satz
    Ergebnis (Punkte)
    Literaturverzeichnis
Infoseite, Layer

(Präsentations-)Stil
(Mathematische) Formel
Bild
Textformate:
Zeilenumbruch, Neuer Paragraph
Zentriert, Zitat, Tabelle, Auflistung/Aufzählung
Textfarbe, Hervorgehobener Text (Def.), Fett, Kursiv
Anker
Eingabe von HTML-formatiertem Text
Literaturverweis, Literaturstelle

Interaktion
Verweise:
zu einem Bild,
zu einer Nebenpfadseite/Infoseite,
auf einen Layer,
zu einem Anker,
der JavaScript ausführt
Aufgaben:
kurze Antwort, Single/Multiple Choice, Hotspot,
Verschiebe-Puzzle (Drag&Drop), Erreichte Punktzahl,
Auswertungsknopf
Applet
MouseOver
Kleiner Hinweis
Eingabefeld (mit Button)
Anzeige einer Formel

MathePrisma Module besitzen eine Struktur ähnlich der eines Buches. Sie gliedern sich in Kapitel und Seiten und jede Seite zerfällt wiederum in Abschnitte.
Die Seiten der Module können entweder linear nacheinander besucht werden oder über die Navigation direkt aufgerufen werden.
Zusätzlich zu diesen Seiten, die den Hauptpfad eines Lernmoduls bilden, gibt es Nebenpfadseiten, die weiterführende oder vertiefende (optionale) Informationen beinhalten und über einen Link von den Hauptpfadseiten aus erreichbar sind.
Ein weiterer Bestandteil von MathePrisma-Einheiten ist das Arbeitsblatt. Dieses kann vom Lehrenden ausgedruckt werden und dient der Lernzielkontrolle.

Es werden drei verschiedene Abschnittsarten unterschieden, die durch eine festgelegte Hintergrundfarbe voneinander abgehoben werden.

- Abschnitte mit motivierendem, erläuterndem oder verbindendem Text (hell grau-blauer Hintergrund)

- Interaktive Abschnitte (weißer Hintergrund)
- Definitionen / (Merk-)Sätze (dunkel grau-blauer Hintergrund)

Die Vorteile dieser eindeutigen Farbwahl und die Stilelemente von MathePrisma werden in Kapitel 6 „Design“ ausführlich beschrieben. In diesem Kapitel befindet sich auf Seite 46 auch eine Abbildung, welche die Oberfläche von MathePrisma zeigt.

Applets bieten bei den Interaktionsformen in MathePrisma den höchsten Grad an Interaktion und werden in allen Modulen häufig eingesetzt (siehe [BKV03]). Sie sind jeweils sehr speziell an die Inhalte eines Moduls angepasst.

Eine Interaktion, der wegen ihrer vielfältigen Einsatzmöglichkeiten besondere Bedeutung zukommt, ist die Diashow (siehe Abbildungen 3.1 und 3.2). Mit Diashows

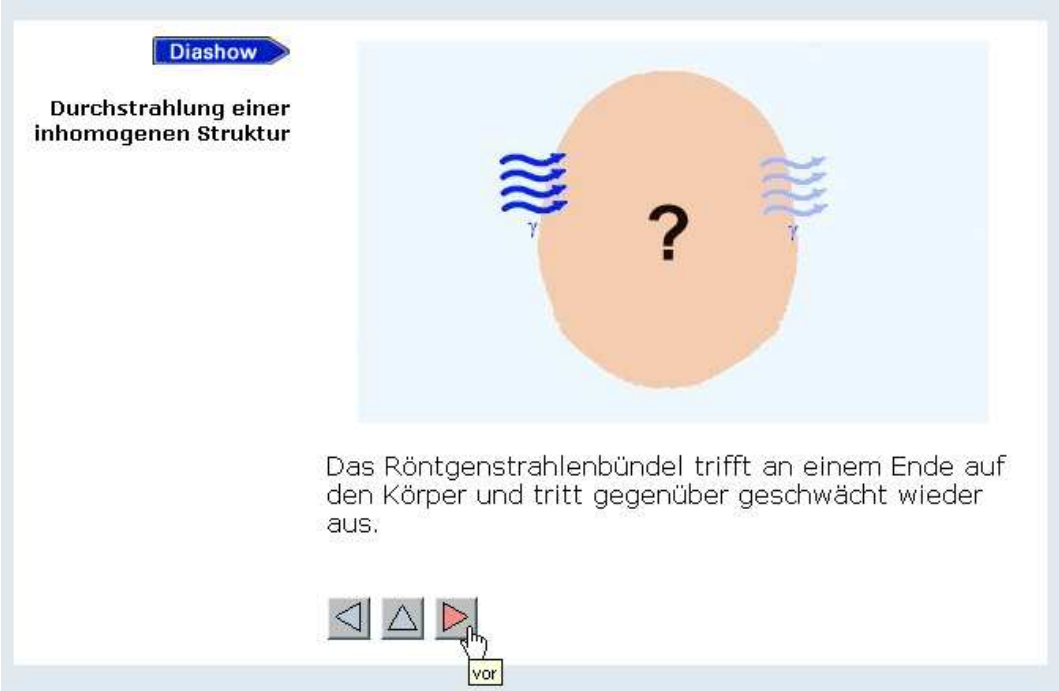


Abbildung 3.1: Diashow aus dem Modul: „CT und Lineare Gleichungssysteme“

können beispielsweise komplizierte Versuchsaufbauten Schritt für Schritt erklärt oder mathematische Beweise nach und nach hergeleitet werden. Sie bieten dabei zwei große Vorteile:

- jedes Dia zieht die Aufmerksamkeit des Lernenden auf sich und wird so intensiv betrachtet,
- der Lernende kann sein Tempo selbst bestimmen und gegebenenfalls „zurückblättern“.

Abbildung 3.2: Diashow aus dem Modul: „Wege auf Graphen“

Es werden zwei verschiedene Arten von Diashows verwendet.
Die einen besitzen drei Buttons: vor, zurück und reset. Wird der vor-Button wiederholt gedrückt zeigen sie ein Dia (Bild) nach dem andern.
Beim zweiten Typ gibt es mehrere Buttons mit beschreibendem Text. Jeder Button ruft nur ein Dia auf.

3.2 Umsetzung in teachTool

Alle Grundelemente von MathePrisma sind in teachTool implementiert. Darüber hinaus ist z.B. eine Verknüpfung zum Herunterladen von Dateien und ein Glossar in der Zusammenarbeit im Projekt OptiV[1] hinzugekommen.

Die in teachTool implementierten Strukturelemente wurden bereits im vorangegangenen Abschnitt beschrieben und ihre Vorteile für E-Learning aufgeführt.

Im Bereich der Grundelemente zur Anpassung des Präsentationsstils ist besonders die Eingabe von Formeln und von HTML-Code hervorzuheben. Ferner können Autoren ihre Literaturhinweise im Literaturverzeichnis verwalten und Verweise darauf im Fließtext setzen. Zur besonderen Herausstellung neu eingeführter Begriffe und

[1] Eine Beschreibung des Projekts OptiV befindet sich in Kapitel 11.

Zusammenhänge gibt es in teachTool eine festgelegte und für den Lernenden leicht zu erkennende Textfarbe.
Wörter, die mit dieser Textfarbe markiert wurden, werden automatisch ins Glossar eingetragen. Bei Konvertieren der Lerneinheit nach HTML wird dieses in der Datei node0.htm angelegt. Mittels dieser Datei ist es möglich, Verweise zwischen zwei Lerneinheiten anzulegen, die auch dann nicht ungültig werden, wenn das Modul, auf das der Verweis zeigt geändert wird. Hebt ein Autor z.B. das Wort „Tangente“ in dem Modul „Ableitung“ hervor, kann er von einem anderen Modul darauf verweisen, indem er folgenden Link als HTML-Code einbaut:

```
<a href="Ableitung-Ordner/pages/node0.htm?TANGENTE"
   target="_blank">Tangente</a>
```

Die Datei node0.htm ruft dann automatisch die entsprechende Seite im Modul „Ableitung“ auf.

Fragen und Aufgaben können nicht nur zur Wissensüberprüfung, sondern auch zum Steuern von Lernprozessen eingesetzt werden. Es wird z.B. in dem MathePrisma-Modul „Primzahlgeheimnisse“ eine Multiple Choice Aufgabe verwendet, um den Lernenden in die Lage zu versetzten, das Prinzip des Siebs des Eratosthenes selbst zu entwickeln (siehe Abbildung 3.3). Oftmals lösen Lernende die Aufgabe, indem sie

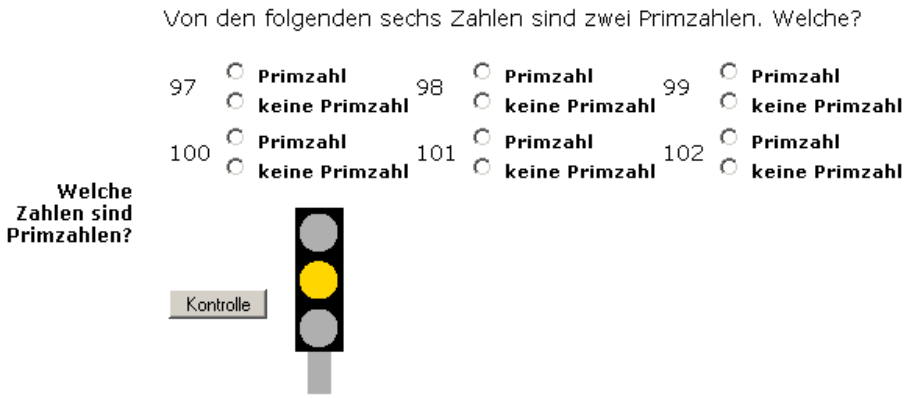


Abbildung 3.3: Aufgabe aus dem Modul: „Primzahlgeheimnisse“

zuerst die Zahlen 98, 100, 102 als nicht prim markieren, da sie durch 2 teilbar sind. Danach kann 99 aufgrund der Teilbarkeit durch 3 als nicht prim erkannt werden. Somit sind nach Aufgabenstellung die verbleibenden Zahlen 97 und 101 Primzahlen. Jeder Lernende, der die Aufgabe so löst, entdeckt dabei in Grundzügen das Prinzip des Siebs des Eratosthenes eigenständig. (siehe [Kri03])
Zur Förderung des entdeckenden Lernens können mit teachTool Aufgaben an jeder Stelle in einem Modul integriert werden. Das stellt einen großen Vorteil gegenüber

den Lerneinheiten der meisten Lernplattformen dar, die lediglich Quizze und Tests anbieten (siehe [Sch03], Seite 84).

Eine Besonderheit bildet das in teachTool integrierte Bewertungskonzept: Bei allen vorgesehenen Aufgabentypen kann der Autor entscheiden, wie viele Punkte der Lernende für die richtige Bearbeitung erhält. Diese Punkte werden in der Lerneinheit automatisch addiert. Es kann für jede Aufgabe entschieden werden, ob die erreichten Punkte angezeigt werden sollen oder nicht. So können auch mehrere Teilaufgaben zu einer größeren Aufgabe gebündelt werden. Am Ende des Moduls gibt es dann die Möglichkeit, eine Gesamtpunktzahl mitzuteilen. Der Autor kann selbstverständlich aber auch ganz auf die Vergabe von Punkten verzichten.

Applets können mit speziellen Programmen beispielsweise mit Dynamischer Geometrie Software (DGS) (z.B. Geonext [Leh]) ohne Programmierkenntnisse erzeugt werden. Diese Programme bieten oft einen HTML-Export, bei dem eine HTML-Datei mit allen für das Applet benötigten Parametern angelegt wird. Der Assistent von teachTool, über den Applets eingebunden werden, besitzt die Möglichkeit, alle benötigten Informationen aus einer HTML-Datei einzulesen und kopiert automatisch die relevanten Dateien (z.B. JAR-Dateien) in den entsprechenden Modulordner.
Auch Applets kann ein Autor mit Punkten versehen und somit in das Bewertungssystem integrieren. Ein entsprechendes Applet muss lediglich die öffentliche Methode `int send_to_html()` besitzen, die -1 zurück gibt, solange die Aufgabe noch nicht bearbeitet wurde und sonst die erreichte Punktzahl. Die Maximalpunktzahl, die vom Applet gemeldet wird, muss dabei mit der in teachTool eingetragenen Punktzahl des Applets übereinstimmen.

Alle Interaktionen, die Aufgaben darstellen, können nur auf Aufgaben-Abschnitten eingesetzt werden. Hat der Autor z.B. einen Abschnitt für normalen Text ausgewählt, sind die entsprechenden Buttons der teachTool-Oberfläche deaktiviert. Auf diese Weise wird die Einhaltung des Designkonzepts auf Autorenseite gewährleistet.

Es gibt einige (Standard-) Interaktionen, wie z.B. Frage-Antwort-Aufgaben, Multiple-Choice oder Diashows, aber auch z.B. Drag&Drop-Aufgaben, deren Erstellung teachTool in ähnlicher Weise durch Wizards bzw. Assistenten unterstützt, wie z.B. das Einfügen von Diagrammen in Microsoft Excel.

Mit teachTool werden Diashows in drei Schritten erstellt:

1. Der Typ der Diashow wird ausgewählt. D.h. der Autor entscheidet, dass entweder die Diashow drei Buttons (vor, zurück und reset) besitzt oder zu jedem Dia ein gesonderter Button mit beschreibendem Text existiert (siehe Abschnitt 3.1). Bei Auswahl des zweiten Typs werden dann in teachTool beliebig viele Buttons mit Textbeschriftung zur Diashow hinzugefügt.

2. Einzelne Bilder oder Bilder mit Texten werden geladen oder erstellt. Sie dienen als Grundbausteine der Dias.

3. Die Diashow kann (einfach) generiert werden, indem alle Bilder (mit Text) nach einander gezeigt werden oder indem der Autor festlegt, wie die Dias aus den Grundbausteinen zusammengestellt werden sollen.

Die Grundelemente sind in teachTool in der Klasse *MPLatexCommands* enthalten. In ihr werden alle Informationen festgelegt, die zum Abspeichern, Einlesen, Anlegen und Bearbeiten der Grundelemente wichtig sind. Als Beispiel wird ein kurzer Teil des Quellcodes im Folgenden erläutert. „commandIndex“ ist darin eine int-Variable, die als Zählindex fungiert und die Position des jeweiligen Grundelements im entsprechenden Array festlegt:

```
type[commandIndex]         = "parano";
parameter[commandIndex]    = new String[]{"Catchphrase"};
vsbParameter[commandIndex]= new String[]{"Catchphrase"};
didParameter[commandIndex]= new String[]{"Time (min.)", "Phase",
                                         "Activity:reading"};
environment[commandIndex] = false;
strToDisplay[commandIndex]= 0;
icon[commandIndex]         = "parano";
text[commandIndex]         = true;
in[commandIndex]           = new String[]{"page", "subpage"};
kind[commandIndex]         = "structure.paragraphs";
tooltip[commandIndex]      = "Normal Text";
```

„parano“ ist ein (normaler) Abschnitt in einer Seite einer Lerneinheit, der motivierenden oder erläuternden Text beinhaltet.
Jeder Seitenabschnitt besitzt eine „Catchphrase“, in der eine kurze Bemerkung oder Zusammenfassung des Abschnitts gegeben werden kann.
Diese Catchphrase ist ein in der Lerneinheit sichtbarer Parameter / sichtbares Attribut von „parano“ und erscheint daher in „vsbParameter“. Alle Parameter die nicht in „vsbParameter“ erwähnt werden, sind automatisch technische Parameter / Attribute.
Didaktische Parameter von „parano“ sind die erwartete Lernzeit des Abschnitts, die Lernphase und die Information für den Autor einer Lerneinheit, welche Art von Aktivität auf Seiten des Lernenden bei diesem Abschnittstyp erwartet wird.
„environment“ ist hier auf „false“ gesetzt und „text“ auf „true“. Das besagt, dass die Information von „parano“ gemäß der Konventionen in LaTeX in folgender Form in einer Datei abgespeichert werden:

```
\parano{Catchphrase-Text}
% DidParam: Time (min.) : erwartete Lernzeit
% DidParam: Phase : Lernphase
```

```
{
Abschnittstext
}
```

Der Text, der im Dokumentstruktur-Baum von teachTool (siehe Abbildung 4.1) für ein „parano“ ausgegeben wird, ist die Catchphrase. Diese ist der Parameter 0.
„in“ beschreibt, welche Knoten im Baum der Dokumentenstruktur als Eltern von „parano“ zugelassen sind. Das sind zum einen eine (Hauptpfad-)Seite und eine Nebenpfadseite. Wird hingegen in teachTool ein Kapitel bearbeitet, bei dem z.B. der Kapitelname geändert wird, so ist der Button zum Einfügen eines „parano“ deaktiviert. Somit wird der Autor dabei unterstützt, nur an geeigneter Stelle Elemente in die Lerneinheit einzufügen.
teachTool besitzt die drei Symbolleisten und Einträge im Menü „Structure“, „Style“ und „Interaction“. Aufgrund des Eintrags in „kind“ wird „parano“ der Symbolleiste „Structure“ und dort dem durch Trennstriche (Separator) festgelegten Bereich „Paragraphs“ zugeordnet.

3.3 Vergleich mit anderen Autorensystemen

Die zum Vergleich mit teachTool untersuchten Autorensysteme ToolBook und Lectora sind ebenso wie teachTool seitenorientiert. Bei ToolBook gibt es jedoch keine Kapitel und somit keine leicht erkennbare hierarchische Strukturierung.
Die Autorensysteme ToolBook und Lectora ermöglichen das Einfügen von Texten und Bildern in ähnlicher Weise wie das Präsentationsprogramm PowerPoint. Der Autor kann Textfelder anlegen und Bereiche für Bilder definieren, die frei auf einer Seite positionierbar sind. Somit besitzt der Autor einen großen Gestaltungsspielraum.
Bei teachTool hingegen gibt es Seiten, denen bestimmte Aufgaben zukommen, z.B. ein Arbeitsblatt und Nebenpfadseiten für das Bereitstellen von weiterführenden Informationen. Außerdem bestehen Seiten stets aus verschiedenen Abschnitten, bei deren Auswahl der Autor berücksichtigen muss, welchem Teil im Lernprozess sie zugeordnet sind. Speziell in der Lehrerausbildung müssen die Studierenden lernen, Unterrichtseinheit zu strukturieren. Es ist beim Aufbau einer Stunde sehr wichtig, die Bedeutung einzelner Phasen wie Problemstellung, Versuche und Sicherung z.B. gemäß dem Stufungsmodell nach Roth (siehe [Mey94]) einschätzen zu können. Diese Art der Vorbereitung erleichtert dem Lernenden das Verständnis des Aufbaus des Unterrichts und damit auch der Inhalte.
Das beschriebene Vorgehen ist nicht nur bei der Gestaltung herkömmlichen Unterrichts wichtig, es sollte auch beim Erstellen von E-Learning-Einheiten gewählt werden. Durch die Vorgabe genau festgelegter Strukturierungselemente wird jedem Autor in teachTool die Wahl eines für E-Learning ungeeigneten Aufbaus nicht nur

erschwert, er wird vielmehr zum Erstellen eines didaktisch geeigneten Aufbau hingeführt. teachTool unterstützt also Autoren, indem es auf sinnvolle Weise den Gestaltungsspielraum einschränkt.

Mathematische Formeln spielen in vielen Fächern eine große Rolle. Sie können in Lectora und teachTool in die Seiten eingefügt werden. ToolBook unterstützt diese Funktion nicht.
Bei der Formatierung von Texten bieten alle drei Autorensysteme Aufzählungen, das Verändern der Textfarbe, Fettdruck und kursiv an. Bei teachTool gibt es zusätzlich noch eine festgelegte Textfarbe, mit der neu definierte Begriffe hervorgehoben werden können. Es wird z.B. im „Didactics Check“ (siehe Kapitel 5) geprüft, ob in einem Abschnitt für Definitionen oder (Merk-)Sätze ein Wort in dieser Farbe markiert wurde. Wenn nicht, erscheint ein Hinweis für den Autor, der ihn auf das Fehlen aufmerksam macht.
Auf der anderen Seite erlauben ToolBook und Lectora das Ändern von Schriftart, Schriftgröße, Hintergrundfarbe und die Formate Unterstrichen und Hochgestellt/Tiefgestellt. Das Einstellen von Schriftart und -größe ist in teachTool nicht vorgesehen, da diese vom Lernenden im Browser individuell gesetzt werden können sollten. Es wird in den Stilvorlagen der HTML-Seiten lediglich eine serifenlose Schrift festgelegt, weil diese am Computer besser zum Lesen geeignet ist (siehe [Dil]). „Da der dargestellte Text auf jedem Browser möglichst homogen und übersichtlich erscheinen soll, müssen Änderungen der Schriftgröße, -stärke, -art und -farbe auf ein Minimum beschränkt werden.“ (siehe [FGK00], Seite 79) Die Hintergrundfarbe der Texte wird in teachTool durch die Wahl des entsprechenden Abschnitts definiert. Auf das Format Unterstrichen wird in teachTool verzichtet, da es durch Fettdruck und Textfarben genügend Möglichkeiten gibt, Begriffe und Zusammenhänge hervorzuheben. Eine Erweiterung der Palette würde die Texte unübersichtlicher machen und bringt keine weiteren Vorteile. Hoch- bzw. Tiefstellen von Texten wird in erster Linie in der Formelumgebung verwendet und ist daher Bestandteil des Formel-Editors.
Das Einfügen von beliebigem HTML-Code versetzt einen Autor in die Lage, individuelle Interaktionen über JavaScript umzusetzen. Diese Funktion bieten die Programme teachTool und Lectora.
Multimediale Elemente wie Videos, Flashdateien und Ton werden in vielen modernen Autorensystemen unterstützt. In diesem Punkt ist eine Erweiterung des Umfangs von teachTool bereits geplant. In Kapitel 11 werden Vor- und Nachteile des Einsatzes dieser Multimediaelemente beschrieben. Als Vorteil sei hier die Verbesserung des Angebots an visuellen Eindrücken, als Nachteil die Abhängigkeit von Plug-Ins genannt.
Die beiden untersuchten Autorensysteme ToolBook und Lectora erlauben das Einfügen einer Vielzahl an Formen und Linien. Da die Text- und Bildkomponenten bei diesen Programmen beliebig platziert werden können, ist dieses Angebot vielseitig einsetzbar. Im Fall von teachTool entfällt jedoch aus den oben beschriebenen konzep-

tionellen Gründen die freie Anordnung. Somit sollte ein Autor Formen und Linien durch Bilddateien integrieren und z.B. mittels Tabellen die Positionierung vornehmen.
Viele Lerneinheiten bieten dem Lernenden einen sequenziell durchlaufbaren Lernpfad an. Das stellt insbesondere für unerfahrene Lernende eine gute Möglichkeit dar, sich zu orientieren (siehe Kapitel 4). Zur Navigation auf diesem (Haupt-)Lernpfad besitzen mit teachTool erstellte Lerneinheiten automatisch Pfeile zum Wechseln zur vorangehenden bzw. zur nächsten Seite und eine Navigationsübersicht. Diese Navigationselemente müssen bei ToolBook und Lectora vom Autor durch Schaltflächen und Aktionen eingebaut werden. In ToolBook kann der Autor ein Template wählen, das entsprechende Buttons standardmäßig auf den Seiten einfügt. Bei Lectora gibt es durch Vererbung die Möglichkeit, die einmal festgelegten Schaltflächen auf allen Seiten erscheinen zu lassen. Es bedarf jedoch weiterer Arbeitsschritte, die ein Autor bei der Konzeption bedenken und bei der Umsetzung ausführen muss. Zusätzlich besitzen teachTool Module im Kopf jeder Seite zur leichteren Orientierung die Angabe von Kapitelname und Seite. Das automatische Einfügen dieser Angabe ist in den beiden anderen Vergleichssystemen nicht vorgesehen.
Mit ToolBook können im Gegensatz zu Lectora und teachTool PowerPoint Präsentationen in Lernmodule integriert werden. Bei der Konvertierung der Lerneinheiten nach HTML wird auf den Internetseiten ein Plug-In zum Anzeigen erforderlich. Die Abhängigkeit von Plug-Ins schränkt jedoch die Plattformunabhängigkeit der Module sehr ein. Es empfiehlt sich an diesen Stellen ein Verweis zum Herunterladen einer entsprechenden Datei.

Im Bereich der Interaktionen haben sich einige Aufgabenformate zum „Standard“ entwickelt. Dazu gehören in erster Linie kurze Antwort und Single/Multiple Choice, aber auch Hotspot und Drag&Drop, die von allen drei untersuchten Autorensystemen bereitgestellt werden. Bei Singlechoice werden von ToolBook und Lectora noch speziell Richtig-Falsch-Aufgaben unterschieden.
Drag&Drop-Aufgaben können in verschieden Ausprägungen eingesetzt werden:

- *zuordnen:* Bilder bzw. Texte besitzen eine eindeutige Zuordnung und werden nebeneinander geschoben, um ihre Zugehörigkeit zu verdeutlichen.
- *anordnen:* Ein Bild ist im Hintergrund sichtbar. Auf diesem sind Bereiche gekennzeichnet, auf die verschiebbare Elemente gezogen werden müssen.
- *klassifizieren:* Es gibt zwei oder mehrere Klassen, denen Bilder bzw. Texte (in beliebiger Reihenfolge) zugeordnet werden.

teachTool besitzt einen Wizard zur Generierung von Zuordnungsaufgaben, die über ein Applet in den HTML-Seiten der Lerneinheiten realisiert werden. Klassifizierungsaufgaben können ebenfalls auf ähnliche Art erstellt werden. Hierfür muss ein Autor jedoch die vom Drag&Drop-Applet verwendete Konfigurationsdatei modifizieren.

ToolBook unterstützt alle Drag&Drop-Formate, Lectora lediglich Anordnungsaufgaben.
Aufgaben zur Eingabe von freiem Text (in beliebiger Länge) können mit Lectora erstellt werden. „Kritisch ist die Antwortanalyse bei freien Texteingaben. Die eingegebene Antwort wird in der Regel nur auf Gleichheit bzw. Exaktheit bzw. auf Synonyme geprüft. Ein Sinnverstehen der Lerner-Eingabe ist nicht möglich, allerhöchstens eine Fehlertoleranz bei Rechtschreibungsfehlern.“ (siehe [Sch02], Seite 105) Diese Aufgaben sind daher zum Selbstlernen, bei dem der Lernende eine unmittelbare Antwort auf seine Eingabe erwartet, nicht geeignet. ToolBook hingegen ermöglicht das Erstellen von Aufgaben bei denen Texte in die richtige Reihenfolge gebracht werden sollen. Matching-Aufgaben, die das Zuordnen von Bildern bzw. Texten durch das Ziehen von Pfeilen bieten, werden von ToolBook und Lectora unterstützt. Diese Art der Aufgabe ähnelt sehr den Zuordnungsaufgaben bei Drag&Drop.
In allen drei Systemen ist vorgesehen, die bei Aufgaben erreichten Punkte zu zählen und dem Lernenden eine entsprechende Auswertung zur Verfügung zu stellen.
Im Projekt MathePrisma haben sich als am häufigsten eingesetzte interaktive Elemente Diashows (siehe oben) und Applets erwiesen. Applets bieten auf plattformunabhängige Weise ein großes Spektrum an Interaktionen. Sie können mit teachTool und ToolBook jedoch nicht (ohne auf HTML-Code zurückzugreifen) mit Lectora eingebunden werden. Diashows werden nur von teachTool angeboten.
MouseOver sind kleine Interaktionen bei denen sich ein Bild verändert, wenn der Lernende mit der Maus darüber fährt. Sie können z.B. sehr gut verwendet werden, wenn sich ein Ausgangszustand aufgrund der Veränderung der Eingangsbedingungen ändert. Auch dieses interaktive Element ist auf direktem Wege nur in teachTool vorgesehen.
Sowohl bei ToolBook als auch bei Lectora gibt es Elemente wie Buttons oder Auswahllisten, die mit bestimmten Aktionen versehen werden können. Gerade bei Lectora ist das Erstellen der Aktionen sehr komfortabel. Es kann aus einer Liste von Zustandsänderungen wie „Button wurde gedrückt“ gewählt und aus einer weiteren Liste eine Aktion zugeordnet werden. Diese individuellen Gestaltungsmöglichkeiten können in teachTool durch die Verwendung beliebigen HTML-Codes umgesetzt werden, bedürfen jedoch guter JavaScript-Kenntnisse.
Als letztes Grundelement dieses Vergleichs sei das Glossar erwähnt, welches bei ToolBook und teachTool verwendet werden kann.

Zusammenfassend kann gesagt werden, dass teachTool alle strukturellen, gestalterischen und interaktiven Grundelemente von Autorensystemen besitzt. In den Bereichen Struktur und Textformatierung ist der Funktionsumfang bewusst eingeschränkt worden, um die Autoren beim Erstellen didaktisch hochwertiger Lerneinheiten zu führen. Insbesondere die Diashow stellt eine wesentliche Bereicherung von teachTool im Gegensatz zu den beiden untersuchten Systemen dar.

Kapitel 4

Struktur

Bei der Strukturierung von Web-Seiten werden oft die folgenden drei Grundmuster berücksichtigt (siehe [LHR99]).

- **Sequenz**
  Eine Sequenz ist eine linear geordnete Abfolge von Seiten, die häufig inhaltlich aufeinander aufbauen.

- **Gitternetz**
  Das Gitternetz ist eine Struktur ähnlich einer Tabelle, in der horizontal und vertikal benachbarte Zellen miteinander verlinkt sind. Die verknüpften Zellen weisen oft ähnliche Charakteristika auf, die als Zeilen- bzw. Spaltenüberschriften der zugrunde liegenden Tabelle angesehen werden können.

- **Hierarchie**
  Einen hierarchischen Aufbau weisen Büchern auf, die eine Gliederung in Teil, Kapitel, Abschnitt usw. besitzen. Auf vielen Internetportalen findet sich eine Hierarchie durch die Bildung von Kategorien, die sich in immer spezieller werdende Kategorien zerlegen.

- **Spinnennetz**
  Als Spinnennetz kann das Internet selbst aufgefasst werden. Es gibt eine Vielzahl von Verknüpfungen, deren Zusammenhang nicht in einer geordneten Struktur dargestellt werden kann.
  Wird dieses Muster zur Gestaltung von E-Learning-Einheiten verwendet, ergibt sich der Vorteil, dass der Lernende sich sehr eigenständig seinen Lernpfad aussuchen kann, ohne einer vorgegebenen Reihenfolge folgen zu müssen. Das wird oft als optimale Umsetzung von konstruktivistischem Lernen angesehen. Großer Nachteil dieser Struktur ist das Problem des „lost in hyperspace“. Ohne jegliche Möglichkeit, sich einen Überblick über die schon besuchten bzw. noch nicht besuchten Seiten verschaffen zu können, ist es gerade für unselbstständige

Lernende schwer sich zu entscheiden, welche Links ausgewählt werden sollen und welche nicht.

Diese Organisationsformen schließen sich nicht gegenseitig aus, sondern können miteinander kombiniert werden und sich bereichern.
Viele Internetsites besitzen beispielsweise eine Überblicksstruktur, die hierarchisch geordnet ist. Es wird zur Navigation auf diesen Seiten häufig auf der linken Bildschirmseite ein Baum angeboten, der die einzelnen Hierarchieelemente darstellt und aufrufbar macht. Die Knoten im Strukturbaum gliedern sich zum Teil selbst weiter auf. Wird ein Knoten ausgewählt, gibt es die Möglichkeit, die dem Knoten untergeordneten Seiten sequenziell zu betrachten. Hierfür werden in der Regel Pfeile nach links und nach rechts angeboten, um die vorherige bzw. nächste Seite der Sequenz aufzurufen.

4.1 Konzept bei MathePrisma

MathePrisma-Module sind so angelegt, dass sie sequenziell bearbeitet werden können. Es gibt hierfür stets in den beiden unteren Bildschirmecken einen Pfeil nach links und nach rechts mit dem vor bzw. zurück gegangen werden kann (siehe Abbildung 6.1). Dieser Vorgehen erweist sich gerade bei Lerneinheiten in der Mathematik als sehr günstig, da die einzelnen Inhalte häufig aufeinander aufbauen.
Zusätzlich weisen MathePrisma-Module durch ihre Gliederung in Kapitel und Seiten eine hierarchische Struktur auf. Es befindet sich daher am unteren Bildschirmrand eine Navigationsübersicht, die diesen Aufbau erkennbar macht und so das unmittelbare Aufrufen von Kapiteln und Seiten ermöglicht.
Neben den Hauptpfadseiten, die sequentiell angeordnet sind, gibt es zusätzlich Nebenpfadseiten mit optionalen, vertiefenden Informationen. Diese Nebenpfadseiten sind über den Hauptpfad erreichbar, wodurch in kleinem Maße die Struktur eines Spinnennetzes realisiert wird. Um das Problem des „lost in hyperspace“ zu vermeiden, kann von den Nebenpfadseiten lediglich zu der Stelle des Hauptpfads zurückgekehrt werden, wo dieser verlassen wurde.
„Umfragen haben gezeigt, dass Benutzer diesen Aufbau nicht als einschränkend, sondern als hilfreich empfinden, da sie so leicht den Inhaltsaufbau überblicken, aber dennoch mittels entsprechender Navigationsmechanismen die Seiten in individueller Reihenfolge bearbeiten können.“ ([Kri03], Seite 41)
Auch die Struktur eines Gitternetzes kann in MathePrisma-Modulen vertreten sein, wenn ein Autor sie für angemessen hält. Im Modul „Sortierverfahren“ werden die Verfahren Sortieren durch Einfügen, Sortieren durch Auswahl, Bubblesort und Quicksort in verschiedenen Kapiteln mit je drei Seiten erläutert. Jedes dieser Kapitel hat einen analogen Aufbau. Auf der ersten Seite eines jeden Kapitels wird die Idee, die hin-

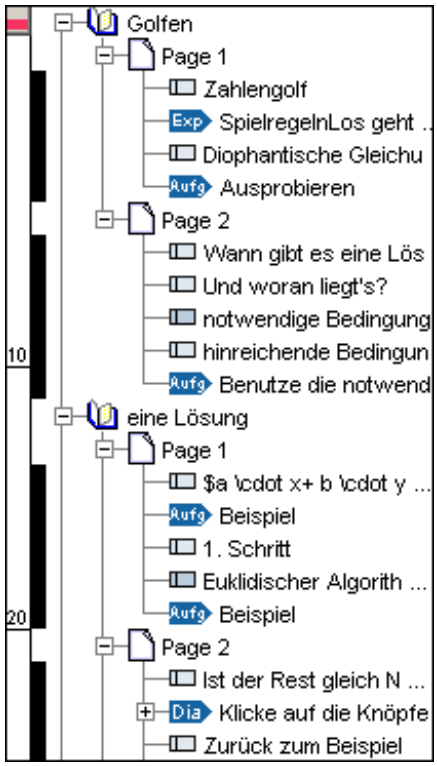


Abbildung 4.1: teachTool Dokumentstruktur-Baum

ter dem Algorithmus steckt, vorgestellt, auf der zweiten Seite wird der Algorithmus Schritt für Schritt erklärt und auf Seite 3 wird der Aufwand analysiert.

4.2 Umsetzung in teachTool

teachTool kann als seitenorientiertes Autorensystem angesehen werden. Jede mit teachTool erstellte Lerneinheit besitzt einen hierarchischen Aufbau, der in Form eines Baums auf der Programmoberfläche dargestellt wird. Die tiefste Ebene bilden dabei Nebenpfadseiten, die als Elternknoten an Hauptpfadseiten angehängt sind.

4.2.1 Vorerfahrungen

Die Navigationsstruktur in den mit teachTool erstellten Lerneinheiten, ist in Anlehnung an MathePrisma sowohl sequenziell als auch hierarchisch. Die Sequenz bietet für Lernende, die alle Inhalte durcharbeiten möchten, oder unerfahrene, unselbstständige Lernende eine leichte Orientierung. Gleichzeitig können andere Lerntypen im Sinne konstruktivistischen Lernens eine freie Auswahl bei der Reihenfolge, der zu bearbeitenden Seiten treffen.

Beim Einsatz von Modulen in der Lehre passiert es häufig, dass ein vorgegebenes Modul nicht perfekt zum Vorwissen der Schüler oder Studierenden oder den gesetzten Lernzielen passt.
So werden z.B. im MathePrisma-Modul „Kombinatorik“ die vier kombinatorischen Grundaufgaben mit Zurücklegen / ohne Zurücklegen, mit Betrachtung der Reihenfolge / ohne Betrachtung der Reihenfolge behandelt. Aufgrund seiner Einsatzerfahrungen hat ein Lehrer die 4. Grundaufgabe (fast) ganz gestrichen und zusätzliche Informationen zum Binomialkoeffizienten hinzugefügt. Auch in Schulbüchern wird der Fall „mit Wiederholung, ohne Berücksichtigung der Reihenfolge“ häufig nicht ausführlich betrachtet, sondern nur in tabellarischer Form (der Vollständigkeit halber) aufgeführt (siehe [GSW90] und [SS91]). Der erwähnte Lehrer hat sich die Mühe gemacht, die HTML-Seiten des Moduls entsprechend anzupassen, was für ihn einen gehörigen Mehraufwand darstellte und einige Kenntnisse in HTML verlangte.

4.2.2 Möglichkeiten in teachTool

Ähnlich wie in der im vorangehenden Abschnitt beschriebenen Situation sehen sich einige Lehrende vor die Aufgabe gestellt, Module auf ihre Bedürfnisse zu adaptieren. Daher können einzelne Abschnitte - aber auch ganze Kapitel - in teachTool mit Drag&Drop ausgetauscht oder je nach Bedarf ganz entfernt werden.
Bei der Auswahl, welche Seiten / Kapitel ausgelassen werden sollen, ist die Zeitleiste am Dokumentstruktur-Baum behilflich. Sie zeigt an, wie viel Bearbeitungszeit einzurechnen ist. Dafür enthält jeder Abschnitt das Metadatum der Bearbeitungszeit, die als Intervall eingegeben wird. Auf Knopfdruck kann die Zeitleiste angepasst werden als Summe der minimalen, maximalen oder durchschnittlichen Zeiten.
Dieses Vorgehen ist natürlich nicht nur bei der Anpassung der Lerneinheit an den jeweiligen Einsatz in der Lehre von großer Bedeutung, es ist auch bei der Erstellung der Lerneinheiten selbst hilfreich.

Zusätzlich zu den Verknüpfungen zu Nebenpfadseiten können mit teachTool Verknüpfungen zu Ankern erstellt werden, die frei in der Lerneinheit positionierbar sind. Sind solche Verknüpfungen vorhanden, eröffnen sie für die Lernenden die Möglichkeit, individuelle Lernpfade auszuwählen. Auch hierdurch wird das Strukturmerkmal eines Spinnennetzes weiter ausgebaut.
Es konnte beobachtet werden, dass diese Verknüpfungen zu Ankern im Projekt OptiV häufiger verwendet wurden als im Projekt MathePrisma.
Das liegt vermutlich zum einen an den jeweiligen Zielgruppen. MathePrisma-Module richten sich in erster Linie an Schüler und Studierende, deren Vorkenntnisse recht bekannt sind. Im Fall von OptiV richten sich die Lerneinheiten verstärkt an bereits Berufstätige. Da deren Vorkenntnisse, z.B. aufgrund unterschiedlicher Berufserfahrung, sehr voneinander abweichen können, bietet sich hier eine intensivere Verzweigung an.

Als weiterer Grund für diese Beobachtung kann das gewählte Fachgebiet gesehen werden. Es überrascht nicht, dass gerade in der Mathematik, in der Inhalte sehr oft aufeinander aufbauen und somit im Projekt MathePrisma ein eher sequenzieller Aufbau gewählt wird.

4.3 Vergleich mit anderen Autorensystemen

Sowohl bei ToolBook als auch bei Lectora erstellt der Autor einzelne Seiten, die über Schaltflächen und Aktionen miteinander verbunden werden. Dadurch können dem Lernenden unterschiedliche Lernpfade angeboten werden. Die erstellte Lerneinheit kann z.B. aufgrund gemachter Lernerfahrungen den Lernenden beim Drücken eines Buttons auf unterschiedliche Seiten bringen. Diese Art des Navigationsangebots bedarf auf Seiten des Autors ein hohes Maß an Planung und Übersicht, um alle sich ergebenden Lernpfade berücksichtigen zu können.
Auf der anderen Seite wurden oben Gründe dafür genannt einen sequenziellen (Haupt-)Lernpfad einzurichten, und an entsprechenden Stellen zusätzliche Informationen über Nebenpfadseiten anzubieten oder weitere Verweise anzulegen. Im Gegensatz zu ToolBook werden in Lectora Lerneinheiten wie bei teachTool anhand eines Strukturbaums erstellt. Es gibt über Vererbung die Möglichkeit auf jeder Seite einen Button zur Wahl der nächsten bzw. der vorangehenden Seite zu erzeugen. Dadurch werden jedoch alle Seiten einer Lerneinheit besucht. Bei teachTool besitzen Nebenpfadseiten die Eigenschaft, dass sie nur besucht werden können, wenn vom Lernenden ein entsprechender Link gewählt wurde. Von ihnen kann auch lediglich auf die Stelle im Hauptpfad zurückgekehrt werden, wo dieser verlassen wurde. Der Autor muss sich bei teachTool nicht um das Erstellen von Elementen zur Navigation kümmern, da Pfeile zum Wechseln auf die nächste bzw. vorige Seite des Hauptpfads automatisch generiert werden. Gleichzeitig wird aber durch Nebenpfade eine Möglichkeit zur Erweiterung von Lernpfaden geschaffen.
teachTool setzt also prioritär auf Sequenz und Hierarchie, da diese Strukturierungsmuster unerfahrene Lerner unterstützen, ohne dabei die Wahl anderer Strukturierungsformen zu unterbinden.

Kapitel 5

Didaktik

Die Wahl eines „angemessenen“ didaktischen Aufbaus ist entscheidend für die Akzeptanz und den Erfolg von E-Learning-Modulen. Was heißt in diesem Zusammenhang „angemessen“?
Die Module müssen insbesondere

1. die Vor- und Nachteile des Mediums berücksichtigen und
2. auf die Zielgruppe abgestimmt sein.

Der größte Vorteil beim Einsatz eines Computers als Lernmedium ist – im Gegensatz zu Printmedien – die Verwendung von Interaktionen. E-Learning-Module sollten daher stets interaktiv gestaltet sein, um (abgesehen von der zeit- und ortsunabhängigen Verfügbarkeit) einen Mehrwert zu erhalten (siehe [Sch03], Seite 151ff.).
Ferner zeigt sich als Problem beim Lernen am Computer, dass die Bereitschaft, lange Texte zu lesen, geringer ist als beim Lesen in einem Buch. Deshalb ist es günstig, Texte auf ihr Minimum zu reduzieren (siehe [Boe02]).
Die Anzahl der zu verwendenden Interaktionen und die zumutbare maximale Länge von Texten hängt dabei stark von der Zielgruppe der Module ab.

5.1 Konzept bei MathePrisma

MathePrisma verfolgt die Absicht, dass sich die Lernenden selbständig und kreativ mit mathematischen Problemstellungen auseinandersetzen. Es ist bei diesem Konzept also nicht nur wichtig, dass eine Vielzahl an Interaktionen vorhanden ist. Diese sollen sich zudem jeweils am Beginn eines neuen thematischen Bereichs befinden. Soweit es möglich ist, werden Definitionen und Theoreme erst dann formuliert, wenn die zugehörigen Begriffe bzw. Aussagen vorher experimentell durch den Lernenden herausgefunden wurden.

Ferner liegt allen MathePrisma-Modulen ein sehr starker Realitätsbezug zugrunde. Probleme und Beobachtungen aus dem Alltag werden aufgegriffen, um Motivationen für das jeweilige Thema und zusätzliche Verankerungen im Gedächtnis herzustellen. Dieser Realitätsbezug kann in erster Linie durch den Einsatz von Visualisierungen erzeugt werden. Hierdurch ist es dann wiederum möglich, das zu Beginn dieses Kapitels genannte Ziel der Minimierung von Texten umzusetzen. Oftmals bietet es sich also an, statt einer Textpassage ein Bild oder eine Diashow zu verwenden.

Die für MathePrisma aufgestellten didaktischen Grundsätze müssen von allen Autoren eingehalten werden. Als Autoren sind an der Universität Wuppertal insbesondere auch Lehramtsstudierende aktiv. Diese erstellen Module im Rahmen ihres 1. Staatsexamens oder in Seminaren, in denen es um Medien- und Vermittlungskompetenz geht. Bei vielen Autoren konnte bei der Erstellung der Inhalte ein Vorgehen in fünf Phasen beobachtet werden, das in [Kri03] und [BKV04] beschrieben wird.

Phase 1: Mathematische Ausarbeitung eines Themas
Vorgehen: Zusammentragen der relevanten mathematischen Definitionen und Sätze unter Berücksichtigung des Vorwissens der Zielgruppe
Resultat: Text- bzw. formelorientierte Ausarbeitung

Phase 2: Konzipierung des Themas
Vorgehen: Anlegen der Phasen: Einstieg, Erarbeitung und Ergebnissicherung; Finden von motivierenden Elementen
Resultat: Strukturierte Inhalte angereichert durch Bilder und Randnotizen

Phase 3: Illustrieren des Themas
Vorgehen: Einfügen von illustrierenden Interaktionen (durch JavaScript und Java), Ergänzen von zusätzlichen Hinweisen auf Extrafenstern
Resultat: Interaktiv illustrierte Inhalte

Phase 4: Unterstützen beim Bearbeiten des Themas
Vorgehen: Einbau von Interaktionen zum Selbstlernen, Umstrukturierung unter handlungsorientierten Gesichtspunkten
Resultat: Inhalte, die zum eigenständigen Lernen anregen

Phase 5: Gestalterische Auseinandersetzung mit dem Modul
Vorgehen: Minimierung der Texte, Herausarbeiten der Kernideen, Erstellung von Inhaltsübersicht und Arbeitsblatt, Prüfen auf didaktisch-stilistische Konsistenz
Resultat: Fertiges MathePrisma-Modul

Dieses Vorgehen lässt sich nach unseren Erfahrungen bei fast allen Autoren – gerade beim Entwickeln ihres ersten Moduls – beobachten.

5.2 Umsetzung in teachTool

Bei den im Projekt MathePrisma beobachteten Bearbeitungsphasen der Lerneinheiten fällt auf, dass die Autoren beim Übergang von einer Phase in die nächste häufig Impulse und Anregungen benötigen. Da diese Hinweise oft strukturell sehr ähnlich sind, ist es sinnvoll, sie als Ratgeber-Funktionalität in teachTool zu integrieren.

5.2.1 Analyse eines MathePrisma-Moduls

Um geeignete didaktische Regeln erstellen zu können, die in teachTool implementiert werden konnten, wurde das MathePrisma-Modul „Ableitungen“[1] auf seinen strukturellen und visuellen Aufbau hin analysiert. Es ist dabei von großer Wichtigkeit, dass in MathePrisma nur drei verschiedene Arten von Abschnitten – unterscheidbar durch die zugeordnete Hintergrundfarbe – existieren: einleitende / verbindende Textabschnitte, interaktive Abschnitte und Definitionen / (Merk-)Sätze.

Es wurden zu den folgenden 5 didaktischen Gesichtspunkten Fragen gestellt, die ein Maß für die Realisierung des jeweiligen Punktes liefern:

- **Visualisierung**
  Wie viele Bilder / Applets[2] wurden auf den Seiten verwendet?
- **Interaktivität**
  Wie viele interaktive Abschnitte befinden sich auf den Seiten?
- **Handlungsorientierung**
  Was geschieht in den ersten drei Abschnitten jeder Seite?
- **Minimierung der Texte**
  Wie viele Wörter befinden sich auf den Seiten / in den Abschnitten der Seiten?
- **Lernaufwand**
  Wie viele Kapitel / Seiten / Abschnitte pro Seite besitzt das Modul?

Die ermittelten Anzahlen zu den Fragen zur Visualisierung, Interaktivität, Minimierung der Texte und z.T. zum Lernaufwand sind pro Seite in Tabelle 5.1 zusammengefasst.

[1] Das Modul „Ableitungen“ wurde gewählt, da es konstant hohe Zugriffszahlen auf dem Web-Server besitzt und in ihm die didaktischen Prinzipien besonders konsequent umgesetzt wurden.

[2] Applets werden hier nur aufgrund ihrer visuellen Eigenschaften genannt, nicht wegen ihrer interaktiven.

	Bilder/ Applets	Interaktive Abschnitte	Wörter	Abschnitte gesamt
Seite 1	7	2	139	3
Seite 2	3	1	68	3
Seite 3	5	2	219	7
Seite 4	2	2	159	3
Seite 5	2	2	215	5
Seite 6	8	4	201	9
Seite 7	3	2	231	6
Seite 8	1	1	103	4
Seite 9	3	4	249	6
Seite 10	1	1	138	9

Tabelle 5.1: Anzahl-Analyse des MathePrisma-Moduls „Ableitungen“

Zur Bestimmung, ob das Modul „Ableitungen“ handlungsorientiert ist, wurde ermittelt, welcher Grad an Interaktivität zu Beginn einer jeden Seite vorhanden ist. Dazu wurden jeweils die ersten drei Abschnitte betrachtet, ob diese einleitenden / verbindenden Text, Interaktion bzw. eine Definition / einen (Merk-)Satz beinhalten (siehe Tabelle 5.2).
8 von 10 Seiten beginnen mit einleitendem Text, in 9 von 10 Fällen kam eine Interaktion unter den ersten drei Abschnitten vor und nur auf 2 der 10 Seiten war unter den ersten drei Abschnitten eine Definition / ein (Merk-)Satz.

Abschnitte	Anzahl auf den Seiten
Text - Interaktion - Text	4
Text - Interaktion - Interaktion	3
Interaktion - Interaktion - Interaktion	1
Text - Definition - Text	1
Definition - Text - Interaktion	1

Tabelle 5.2: Abschnitts-Analyse des MathePrisma-Moduls „Ableitungen“

5.2.2 In teachTool implementierte Regeln

Als Ergebnis der Analyse und unter Berücksichtigung der für MathePrisma-Module geltenden Vorgaben (vgl. z.B. Style Guide[3]) wurden die auf der nächsten Seite folgenden, in vier Kategorien aufgeteilten Regeln aufgestellt.

[3] http://www.matheprisma.de/Module/TUTORIUM/StylGuid/index.htm

Modul Regeln

M1 Ein Modul muss mindestens 4 Kapitel besitzen.

M2 Ein Modul darf höchstens 6 Kapitel besitzen.

M3 Ein Modul darf höchstens 16 Seiten besitzen.

M4 Ein Modul muss mindestens 4 Aufgaben auf dem Arbeitsblatt besitzen.

Kapitel Regeln

K1 Jedes Kapitel darf höchstens 5 Seiten besitzen.

Seiten Regeln

S1 Jede Seite beginnt mit einem (einführenden) Textabschnitt.

S2 Eine Definition / ein (Merk-)Satz darf nicht unter den ersten 2 Abschnitten vorkommen.

S3 Ein interaktiver Abschnitt muss unter den ersten 3 Abschnitten sein.

S4 Auf einer Seite dürfen höchstens 9 Abschnitte sein.

S5 Auf einer Seite dürfen höchstens 3 Definitionen / (Merk-)Sätze sein.

S6 Auf einer Seite dürfen höchstens 250 Wörter sein.

S7 Jede Seite muss ein Bild oder Applet besitzen.

Abschnitts Regeln

A1 In einem Abschnitt dürfen höchstens 100 Wörter sein.

A2 Jede Definition / jeder (Merk-)Satz muss ein hervorgehobenes Wort besitzen.

A3 Jeder interaktive Abschnitt muss eine Interaktion besitzen.

Die Regeln M2 und K1 finden sich annähernd in [Kri03] auf Seite 57: „Dabei sollte ein Modul nicht mehr als fünf Kapitel und jedes Kapitel nicht mehr als fünf Seiten beinhalten [...].“
S. Krivsky begründet diese Einschränkung durch eine aus Erkenntnissen der Kognitionspsychologie gewonnene Forderung:
„Forderung 15: Einzel-Informationen sind zu bündeln, da sie so deutlich besser vom

KZG[4] aufgenommen werden. Bei 7(+/-2) 'Einheiten' ist die Aufnahme am ehesten gewährleistet.“ ([Kri03], Seite 21)

Ähnliche quantitative Strukturmerkmale – wie sie durch M1 und M2 beschrieben werden – lassen sich auch im Internetprojekt MaDiN [Ste] beobachten, in dem unter Verwendung einer Objektorientierten Themenanalyse (siehe [Ern05]) auf Strukturierung besonders Wert gelegt wurde. Das ist von besonderem Interesse, da MaDiN aufgrund seiner mathematikdidaktischen Inhalte in erster Linie für Lehramtsstudierende entwickelt wurde und daher nicht unbedingt die Bedürfnisse von Schülern erfüllen muss, wie es bei MathePrisma der Fall ist. Das MaDiN-Modul „Didaktik der Algebra“ (Verantwortlicher laut der MaDiN-Internetseiten: H.-G. Weigand) beispielsweise gliedert sich in die 5 Abschnitte „Algebra in der Schule“, „Zahlen“, „Terme“, „Funktionen“, „Gleichungen“, die selbst wiederum in 3 bis 5 Unterabschnitte zerfallen. Diese Einteilung entspricht der durch M1 und M2 vorgegebenen Spanne von 4 bis 6 Kapiteln bzw. Abschnitten.

Als Höchstzahl für die Seiten eines Moduls würde sich als Produkt aus höchstens 5 Kapiteln und höchstens 5 Seiten pro Kapitel 25 ergeben. Diese Zahl erweist sich jedoch für Lernende, die ein Modul von vorne bis hinten durcharbeiten möchten, als zu groß und damit demotivierend. Die Stand 11.01.2006 sich im Internet befindenden 39 MathePrisma-Module besitzen im Durchschnitt weniger als 12 Seiten. Nur 7 Module bestehen aus mehr als 14 Seiten. Daher erscheint eine „Richtwert“ von höchstens 16 Seiten als angemessen. Eine Einschränkung der Anzahl an Abschnitten pro Seite dient sowohl der Eingrenzung des Lernumfangs auf ein überschaubares und damit motivierendes Maß. Gleichzeitig wird dadurch das Scrollen auf den Seiten verringert, das im Bereich des Web-Design und von Benutzern zum Teil als störend empfunden wird.

5.2.3 Vergleich verschiedener Altersstufen

Das Modul „Ableitungen“ behandelt ein Thema der schulischen Oberstufe. Es ist daher interessant zu untersuchen, ob sich Abweichungen zu den obigen Regeln ergeben, wenn ein Modul für eine andere Altersstufe betrachtet wird.
Das MathePrisma-Modul „Klecksaufgaben“ beispielsweise beschäftigt sich mit einem Mittelstufenthema. Eine Analyse diese Moduls ergab eine Bestätigung der meisten der aufgestellten Regeln. Das Modul besitzt z.B. 4 Kapitel, 14 Seiten und höchstens 4 Seiten pro Kapitel.
Es stellt sich nur die Frage, ob die Maximalzahlen für Wörter pro Seite / pro Abschnitt für diese Altersstufe heruntergesetzt werden sollten. Die höchste in diesem Modul vorkommende Anzahl Wörter pro Seite ist zwar 233, was nah an der oben gewählten Höchstzahl 250 liegt (siehe S6). Alle anderen Seiten besitzen jedoch nicht

[4] Kurzzeitgedächtnis

mehr als 152 Wörter. Ähnlich verhält es sich bei den Wörtern pro Abschnitt. Im Modul „Klecksaufgaben“ gibt es höchstens 66 Wörter pro Abschnitt, ein Wert der einiges unter dem von 100 in A1 liegt.
Auch die Regel S5: „Auf einer Seite dürfen höchstens 3 Definitionen / (Merk-)Sätze sein.“ sollte für Module der Mittelstufe evtl. etwas angepasst werden. Im Modul „Klecksaufgaben“ gibt es lediglich einen Merksatz.

Die Lerneinheiten, die im Projekt OptiV (siehe Kapitel 11) erstellt wurden, wenden sich in erster Linie an erwachsene Lernende. Es werden daher häufig mehr als die für MathePrisma-Module empfohlene maximale Anzahl an Wörtern verwendet. Aufgrund der Zielgruppe ist es angemessen, die Zahlen in S6 und A1 für OptiV zu erhöhen.

5.2.4 Der „Didactics Check“

teachTool ermöglicht die Überprüfung aufgestellter didaktischer Regeln mit Hilfe des „Didactics Check“.
Die zu überprüfenden Regeln können über „Didactics Check Options“ (siehe Abbildung 5.1) eingestellt werden. Es kann im entsprechenden Optionenfenster z.B. „MathePrisma-Rules“ ausgewählt werden. Dann findet die Überprüfung des Lernmoduls unter Berücksichtigung der oben genannten Regeln statt. Wird im Optionenfenster „User defined-Rules“ ausgewählt, so können die vorgegebenen Regeln modifiziert oder außer Funktion gesetzt werden.
Wenn der Button „Didactics Check“ gedrückt wird, erscheint ein Bericht, in dem alle Regelverletzungen aufgelistet sind. Der Autor erhält die Information, welche Regel wo übertreten wurde und einen Hinweis, wie er das Problem beheben kann. Abbildung 5.2 zeigt ein Beispiel für einen solchen Bericht.

Neben diesen Mechanismen, welche die strukturelle Beschaffenheit eines Moduls überprüfen, muss, wie schon in Kapitel 3 angedeutet, auch die sinnvolle Verwendung der Abschnittstypen sichergestellt werden (siehe A2 und A3). Ein interaktiver Abschnitt am Anfang einer Seite ist schließlich nur dann ein Indiz für einen handlungsorientierten Aufbau, wenn er auch tatsächlich Möglichkeiten zur Interaktion liefert und nicht lediglich ein Text-Abschnitt mit einer anderen Hintergrundfarbe ist. Es wird von teachTool erkannt, ob z.B. ein interaktiver Abschnitt tatsächlich eine Aufgabe oder ein Experiment besitzt. Ist das nicht der Fall, wird dem Autor vorgeschlagen, eine entsprechende Ergänzung vorzunehmen.

5.2.5 Unterstützung von teachTool beim Erstellen von E-Learning

Wie kann teachTool den Autor bei der Bewältigung der auf Seite 34 dargestellten fünf Phasen bei der Erstellung von E-Learning-Einheiten unterstützen?

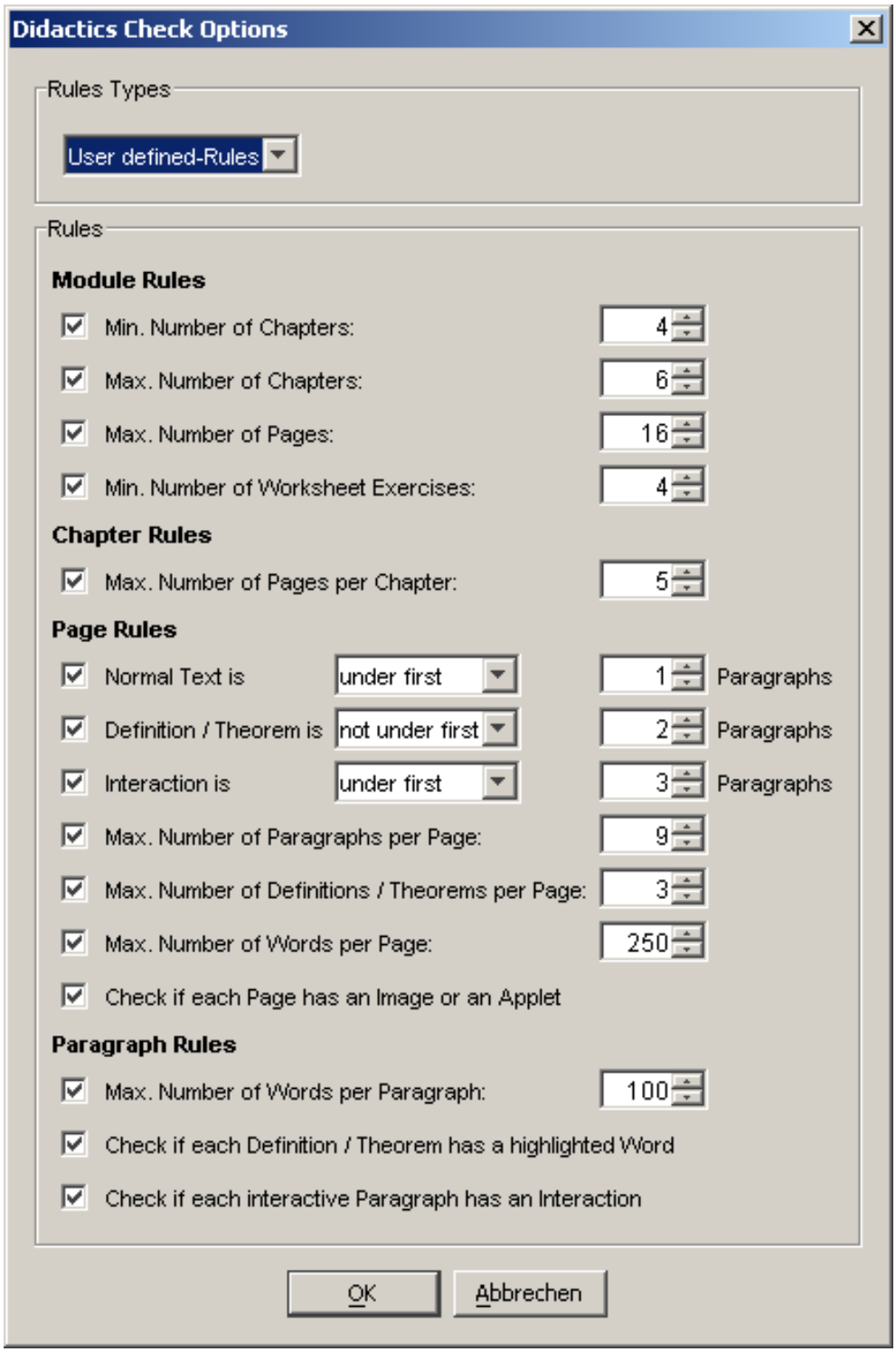


Abbildung 5.1: teachTool „Didactics Check“ Optionen

Seite	Kapitel		Problem	Vorschlag
2	Verschlüsseln	1	Der Abschnitt 2 auf dieser Seite ist ein(e) Definition/Satz. Definitionen/Sätze sollten frühestens als 3. Abschnitt vorkommen.	Setzen Sie den Abschnitt an eine Position weiter unten.
2	Verschlüsseln	1	Es gibt keinen interaktiven Abschnitt unter den ersten 3 Abschnitten.	Erstellen Sie einen interaktiven Abschnitt, Exp, Aufg, Dia, oben auf der Seite oder verschieben Sie einen nach oben.
3	Verschlüsseln	2	Am Anfang dieser Seite befindet sich kein einleitender Text.	Fügen Sie einen Text-Abschnitt am Anfang der Seite ein.

Abbildung 5.2: teachTool „Didactics Check“ Hinweise

In *Phase 1*, in der es in erster Linie um das Zusammenstellen des fachlichen Wissens geht, kommt in mathematischen und naturwissenschaftlichen Gebieten der in teachTool eingebaute Formel-Editor zum Einsatz (siehe Kapitel 7).
Das Anlegen und Aufteilen in (Unterrichts-)phasen findet in *Phase 2* statt. Bezogen auf den Gesamtaufbau des Moduls regt die „Didactics Check“ Regel M4 dazu an, mindestens vier Kapitel anzulegen, die Einstiegsphase, Erarbeitungsphase und Ergebnissicherung (siehe [Mey94]) abdecken sollten.
Mehr im Detail kann in teachTool für jeden Abschnitt einer Seite festgelegt werden, zu welcher (Unterrichts-)phase er gehört. Unterschieden werden: Einstiegs-, Erarbeitungs-, Auswertungs-, Kontroll- und Vertiefungsphase. Wird einem Abschnitt eine Phase zugeordnet, erscheint im Dokument-Strukturbaum vor diesem Abschnitt ein Rechteck in einer der Phase zugeordneten Farbe (siehe Abbildung 5.3). Wenn im Strukturbaum also sehr häufig die Farbe vor den Abschnitten variiert, ist das ein Zeichen dafür, dass die Phaseneinteilung noch nicht zufriedenstellend ist. Der Autor kann dann z.B. durch Drag&Drop Abschnitte in eine andere Reihenfolge bringen.
Auch die Regel S1: „Jede Seite beginnt mit einem (einführenden) Textabschnitt.“ kann in Phase 2 dabei unterstützen, einen geeigneten Aufbau zu finden.
Weiterhin werden in dieser Phase Visualisierungen in das Modul eingefügt, was durch S7 unterstützt wird.
Phase 3 beinhaltet das Erstellen von Interaktionen. Dazu wird der Autor durch S3 aufgefordert. Wie schon vorher erwähnt kann jedoch eine Überprüfung auf die Verwendung interaktiver Abschnitte nur als aussagekräftig angesehen werden, wenn die interaktiven Abschnitte auch tatsächlich Interaktionen beinhalten. Es spielt hier also auch Regel A3 mit herein.
Eine handlungsorientierte Ausrichtung der Lerneinheit, wie sie in *Phase 4* umgesetzt

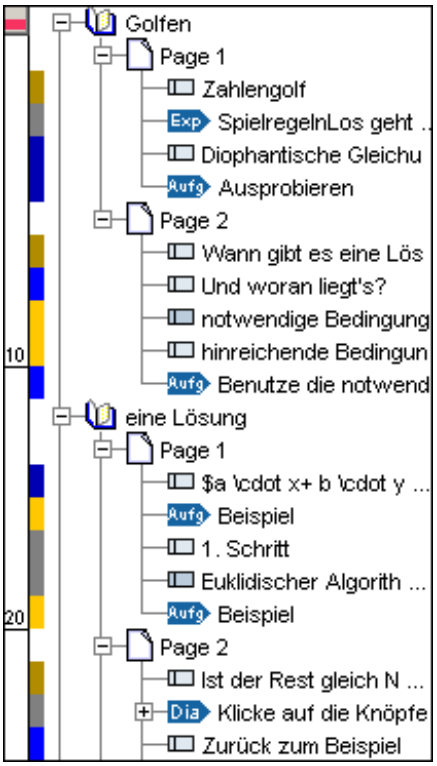


Abbildung 5.3: teachTool Dokumentstruktur-Baum mit zugeordneten Phasen

werden soll, wird ebenfalls durch S3 und A3 angeregt. Zusätzlich ist es wichtig den Stellenwert von Definitionen / (Merk-)Sätzen geeignet zu wählen. Dies unterstützen S2 und S5.
In der *Phase 5* wird der Text auf ein Minimum reduziert (S6 und A1), das Arbeitsblatt angelegt (M4) und sichergestellt, dass Designvorgaben erfüllt sind (A2). Ferner muss nochmals dafür gesorgt werden, dass kognitionspsychologische Grundregeln erfüllt werden, die einen sowohl fordernden als auch motivierenden Lernaufwand gewährleisten sollen (M2, M3, K1, S4).

Eine weitere Funktionalität in teachTool ist die „Didactics Documentation“. Sie gibt in Berichtsform die grundlegenden didaktischen Merkmale einer Lerneinheit wie Zielgruppe, Bearbeitungszeit und Lernziel wieder. In tabellarischer Form wird eine Gliederung des Moduls angegeben, die durch die vom Autor festgelegte Zuordnung der Seitenabschnitte zu entsprechenden Lernphasen bestimmt ist. Die Spalten der Tabelle umfassen die Lernzeit, den Namen der Phase, die beteiligten Kapitel inklusive ihrer Lernziele und die in den Kapiteln enthaltenen Formen von Aktivitäten.

Durch die Implementierung des „Didactics Check“ und der „Didactics Documentation“ stellt teachTool als Autorensystem nicht nur eine Lehrsoftware dar. Es kann vielmehr auch – insbesondere in der Lehramtsausbildung – als Lernprogramm auf dem Gebiet der Fachdidaktik verwendet werden.

5.3 Vergleich mit anderen Autorensystemen

Die Autorensysteme ToolBook und Lectora bieten keine Funktionalität, die den Autor beim Einhalten didaktischer Regeln unterstützt bzw. deren Überprüfung ermöglicht.
Daher wird das Programm LERSUS [DEL] in Version 3.2 zum Vergleich mit teachTool herangezogen. Es ist das einzige dem Autor bekannte Autorensystem, das im Bereich der didaktischen Benutzerführung zu teachTool vergleichbare Funktionen besitzt.
Wenn mit LERSUS eine neue Lerneinheit angelegt wird, muss der Autor sich für ein zugrundeliegendes „didaktisches Modell“ entscheiden. Dieses Modell legt, laut der Internetseite des Herstellers [DEL], Regeln für die folgenden Bereiche fest:

- „Anforderungen an Struktur des Dokuments
  - Beschreibung der Kapitel
  - Beschreibung der Absätze
  - Regeln der Zusammenarbeit von Elementen im Dokument
  - Regeln der Transformation von Elementen im Dokument
- Einstellungen der Editorfunktionen
- Stilarten und Design für visuelle Darstellung des Dokuments
- Existierende Exportformate
- Regeln für Interpretation des Moduls beim Export“

Die „didaktischen Modelle“ sind XML-Dateien, die z.B. im Fall des Standardmodells „easyContent“ folgende Struktur einer Lerneinheit vorschreiben:

- Modul
  - Metadaten
  - Kapitel
    - Beispiel
    - Übung
    - Interaktion
  - Test
  - Referenzen

Es gibt bei LERSUS ausgewiesene Komponenten, wie Beispiel, Übung oder Interaktion. Fügt ein Autor eine Komponente Übung in einem Kapitel nach einer Interaktion ein, erscheint beim Überprüfen der Lerneinheit aufgrund des Modells „easyContent“ eine Meldung, dass die Vorgaben verletzt wurden.

Mit LERSUS erstellte Module können nach HTML und in ein SCORM-konformes (siehe Kapitel 8) Format exportiert werden. Die „didaktischen Modelle“ legen ebenfalls fest, wie die Hintergründe und der Aufbau der exportierten Internetseiten aussehen und welche SCORM-Version erfüllt wird.

Die in LERSUS gewählte Methode einer didaktischen Benutzerführung stellt eine gute Grundlage dar, den Autor für strukturelle Anforderungen zu sensibilisieren.
Es ist einem Autor jedoch nur mit einigem Aufwand möglich, die aufgestellten didaktischen Regeln zu modifizieren. Zwar werden dazu die beiden Werkzeuge LERSUS Model Editor und LERSUS Model Wizard zur Verfügung gestellt. Diese benötigen jedoch einige Eingewöhnungszeit und bilden in erster Linie die XML-Struktur der Modell-Dateien ab. Das Verändern der Modelle sollte insbesondere deshalb innerhalb des Programms angeboten werden, da das Standardmodell „easyContent“ durch seine vorgegebene Strukturreihenfolge Beispiel, Übung, Interaktion kein handlungsorientiertes Schema verfolgt.
Ferner ist die Überprüfung der Struktur etwas oberflächlich. Es wird nicht wie bei teachTool sichergestellt, dass eine interaktive Komponente auch tatsächlich Interaktionen beinhaltet. Es ist den Autoren bei LERSUS also möglich, alle Strukturmerkmale zu erfüllen, ohne dadurch eine didaktische Vorgabe zu erfüllen.
Keine Möglichkeit bietet LERSUS – im Gegensatz zu teachTool – Lerneinheiten auf die Verwendung von Visualisierungen, den Umfang der Texte bzw. den Lernaufwand hin zu überprüfen.

Kapitel 6

Design

Wenn mehrere Internetseiten zu einem Projekt oder Lernmodul zusammengehören, empfiehlt es sich, ein einheitliches Design („Corporate-Design“) zu verwenden. Dieses macht das Betrachten der Seiten einerseits vom ästhetischen Gesichtspunkt her ansprechender. Viel wichtiger im Hinblick auf Lernmodule ist jedoch, dass durch ein einheitliches Design dem Lernenden die Orientierung wesentlich erleichtert wird.

6.1 Konzept bei MathePrisma

Jedes MathePrisma-Modul besitzt eine Startseite mit einem Foto, das dem alltäglichen Leben entnommen ist und gleichzeitig das Thema des Moduls widerspiegelt. „Zum einen macht ein solches Foto neugierig auf den Inhalt, motiviert also, zum anderen kann dadurch die Idee eines Moduls leichter im Gedächtnis verankert werden, da es an Bekanntes anknüpft. Auch bringt sich die Mathematik dadurch immer wieder ins Gedächtnis, wenn entsprechende Gegenstände im Alltag wieder erkannt werden [...].“ ([Kri03], Seite 58) Das Foto trägt dazu bei, dass der Lernende eine innere, persönliche Begründung für den Lernstoff findet, wie es in [BB96] gefordert wird. Außerdem zeigt eine Startseite die Informationen: Titel des Moduls, Untertitel, Namen der Autoren und den Monat der Erstellung.

Im Falle von MathePrisma gibt es stets eine Navigationsleiste am unteren Bildschirmrand (siehe Abbildung 6.1). „Wie sich in Testphasen von MathePrisma gezeigt hat, animiert die Leiste am oberen Rand zu schnell zum ’Weiterklicken’ und konkurriert sowohl am rechten als auch am linken Rand zu sehr mit dem jeweiligen Seiteninhalt.“ ([Kri03], Seite 57)

„Die Kopfzeile jeder Seite setzt sich zusammen aus dem MathePrisma-Logo, dem (verkleinerten) Bild der Startseite, sowie dem Titel des Moduls, dem Namen des Kapitels (in Klammern) und der Seitennummer innerhalb des Kapitels (hochgestellt) bzw. dem Namen der Nebenpfadseite (hochgestellt).“ ([Kri03], Seite 60)

Die Hintergrundfarbe der einzelnen Abschnitte auf den Seiten eines Moduls bereitet den Lernenden durch einheitliche Farbwahl darauf vor, ob er

- eine Interaktion (weiß),
- normalen (erläuternden oder einleitenden) Text (hellblau)
- oder eine Definition / einen (mathematischen) Satz (dunkelblau)

zu erwarten hat. In Abbildung 6.1 ist eine Seite aus MathePrisma zu sehen, die das Design widerspiegelt, das insbesondere mit Blick auf Regeln aus dem Kommunikations-Design[1] entwickelt wurde.

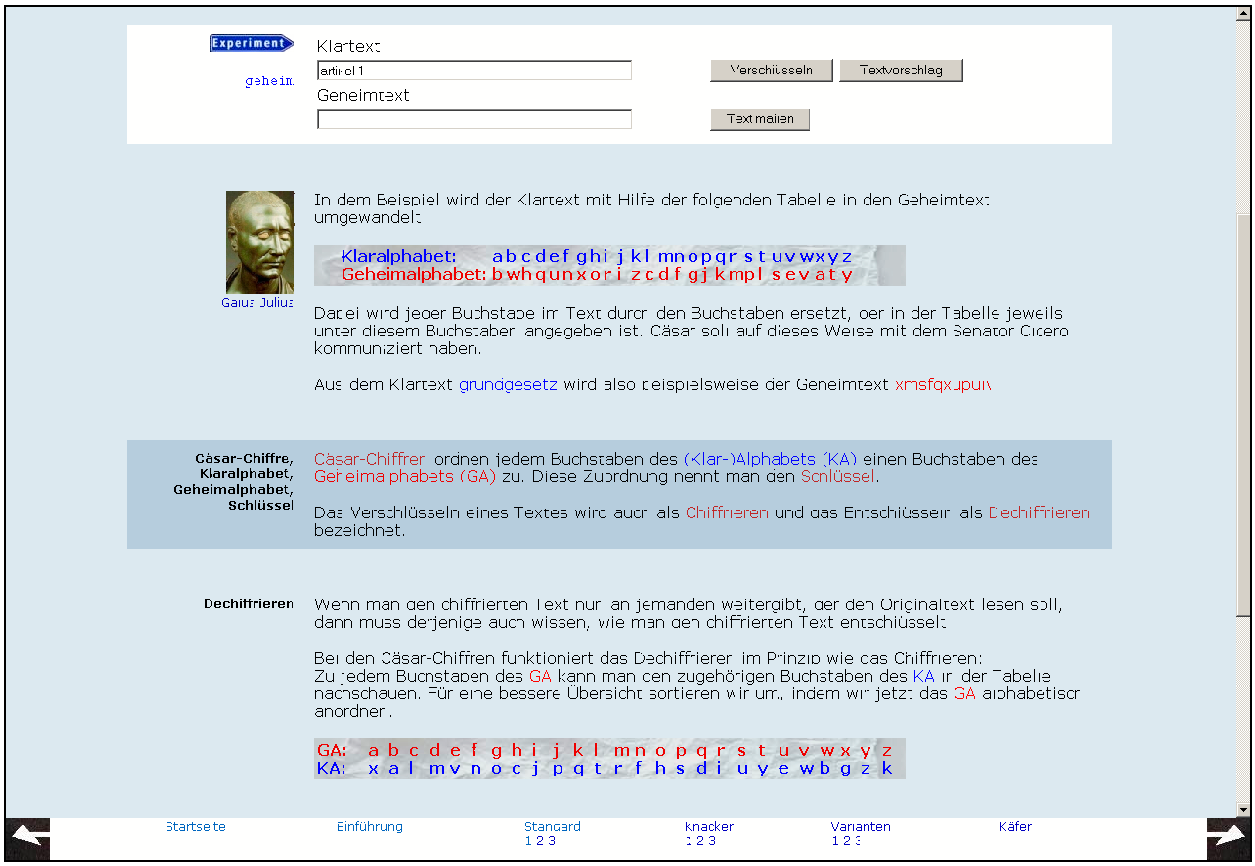


Abbildung 6.1: MathePrisma Abschnitte aus dem Modul: „Cäsar-Chiffren“

[1] Das Erscheinungsbild von MathePrisma wurde von der (Kommunikations-)Designerin N. Bölt überarbeitet.

6.2 Umsetzung in teachTool

teachTool bietet vielfältige Möglichkeiten, auf das Design einer Lerneinheit Einfluss zu nehmen. Sie wirken sich aus, wenn die Lerneinheit nach HTML konvertiert wird. Verschiedene Einstellungen können über das Fenster „Design Options“ gewählt werden (siehe Abbildung 6.2).

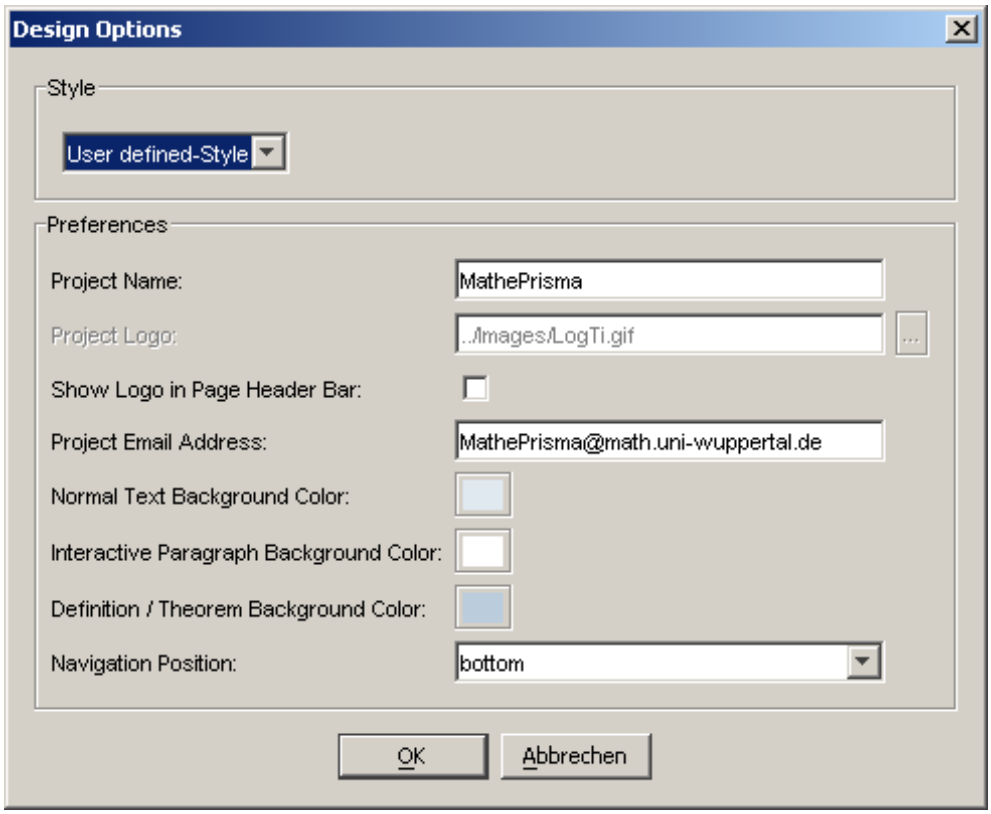


Abbildung 6.2: teachTool Design Optionen

Da sich sowohl das didaktische Konzept als auch das Design von MathePrisma in den letzten Jahren sehr bewährt haben (siehe [Kri03]), orientiert sich teachTool an diesen Vorgaben.

Die für die Startseite benötigten Informationen werden in teachTool in einem Fenster eingegeben, das dem Autor während der Eingabe sofort das Aussehen der Seite im Browser anzeigt (siehe Abbildung 6.3).

Die Navigationsleiste wird aufgrund der Informationen über die einzelnen Kapitel und Seiten automatisch erzeugt. Es kann eingestellt werden, an welcher Position (links oder unten) sie sich befinden soll. Es wurde die Möglichkeit in teachTool implementiert, die Navigationsleiste auf die linke Bildschirmseite zu platzieren, da sich diese Position auf vielen Internetsites zum Standard entwickelt hat.

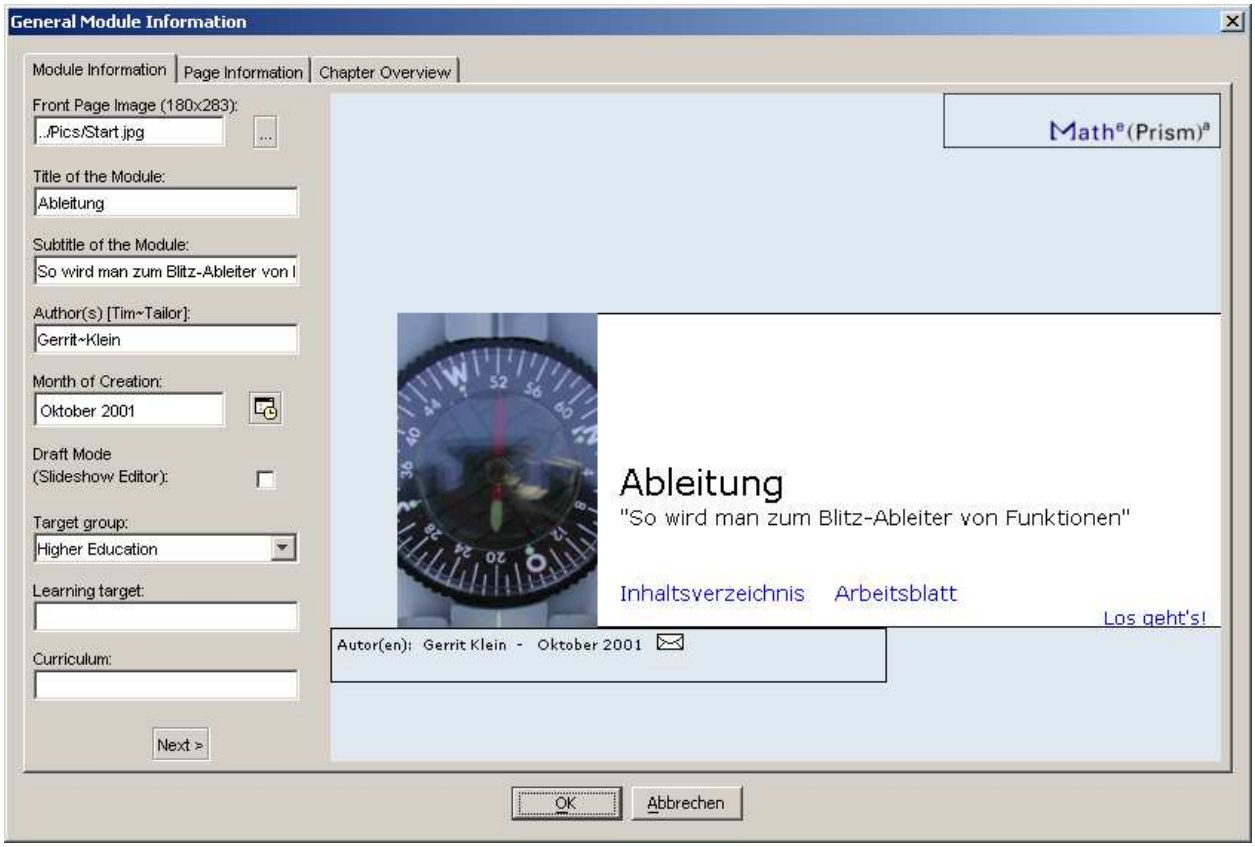


Abbildung 6.3: Erstellen einer Startseite mit teachTool

Trotzdem wurde wegen der im vorangehenden Abschnitt genannten lerntheoretischen Erkenntnisse auch im Projekt OptiV eine Navigationsleiste am unteren Bildschirmrand gewählt.

Die Kopfzeile jeder Seite beinhaltet bei MathePrisma das Projekt-Logo (Höhe 50 Pixel) oberhalb eines weißen Balkens der Höhe 140 Pixel. Im Projekt OptiV wurde hingegen ein Logo mit der Höhe 107 Pixel gewählt. Es wird daher in den teachTool Design Optionen angeboten, das Projekt-Logo in den Balken des Seitenkopfs einzubetten, um die Höhe der Kopfzeilen zu verringern (siehe Abbildung 6.4).

Es können bei teachTool nicht beliebige Abschnitte in beliebigen Farben in die einzelnen Seiten eingebaut werden. Der Autor muss sich stets vor dem Einfügen Gedanken machen, welche Funktion ein Abschnitt erfüllen soll. Hat er sich z.B. dafür entschieden, eine Interaktion zu integrieren, muss (bzw. kann) er sich keine weiteren Gedanken über die zu wählende Hintergrundfarbe machen. Diese kann jedoch einheitlich für das gesamte Modul individuell angepasst werden.

Diese Art des Vorgehens erfordert auf Seiten des Autors einen (kleinen) zeitlichen Mehraufwand. Sie zwingt ihn dazu, erste wichtige Strukturierungen in seinem Modul vorzunehmen und sich dabei über die jeweilige Rolle eines Abschnitts klar zu

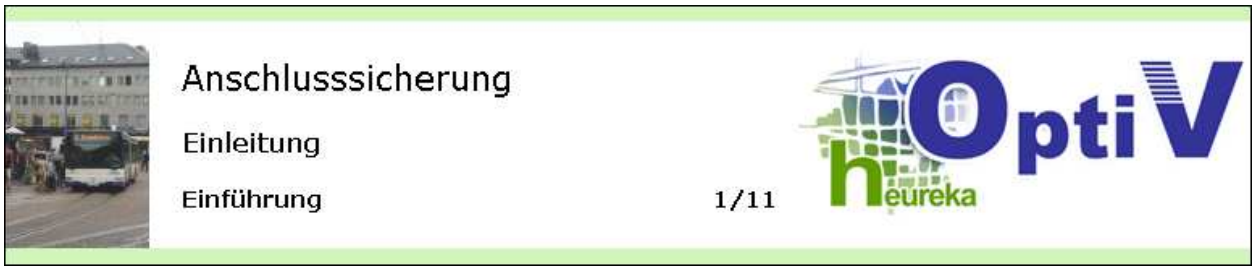


Abbildung 6.4: Kopfzeile einer Seite aus der OptiV-Lerneinheit „Anschlusssicherung“

werden.
Auf Seiten des Lernenden wird eine wichtige Orientierung gefördert; er kann so auch nicht durch eine sinnlose Farbenvielfalt verwirrt oder abgelenkt werden (siehe [Kri03], Seite 60-62). Darüber hinaus werden die Abschnitte in einem Zwei-Spalten-Layout, bei dem die linke Spalte 18% und die rechte Spalte 82% einnimmt, so untereinander auf den Seiten angeordnet, dass sich gut erkennbare Orientierungslinien ergeben. Viele Abschnitte werden automatisch mit entsprechenden Icons versehen, die zusätzlich die Funktion der Inhalte (z.B. Aufgabe oder Experiment) anzeigen.

6.3 Vergleich mit anderen Autorensystemen

Wie bei PowerPoint kann bei ToolBook und Lectora ein Rahmen und ein Hintergrund gewählt werden, der auf allen Seiten einer Lerneinheit angewendet wird. Dadurch ist eine leichtere Überblickbarkeit und ein zusammenhängendes Äußeres im Sinne eines „Corporate-Designs“ realisierbar.
Die Texte werden jedoch auf den Seiten in Textfeldern eingefügt, deren Hintergrundfarben frei variierbar sind. Macht ein Autor davon häufig Gebrauch, werden die Seiten sehr leicht mit Farben überfrachtet. Außerdem kann die freie Positionierbarkeit der Textfelder dazu führen, dass die Seiten keine einheitlichen und gut erkennbaren Orientierungslinien besitzen und damit unübersichtlich sind. Es gibt zwar bei beiden Programmen Funktionen, um die Bestandteile der Seiten auszurichten. Diese setzen aber bei den Autoren Kenntnisse im Kommunikations-Design voraus, die vermutlich nicht immer vorhanden sind.

Des Weiteren ist eine feste Zuordnung zwischen Hintergrundfarbe und Funktion eines Textes, wie oben beschrieben, sehr unterstützend für den Lernprozess. Diese Zuordnung muss ein Autor bei ToolBook und Lectora selbstständig vornehmen und auch konsequent umsetzen.

Gerade die große Wahlfreiheit im Bereich des Designs erschwert bei ToolBook und Lectora dem Autor die Arbeit, da er sehr gutes Vorwissen in Kommunikations-Design und Mediendidaktik besitzen und stets berücksichtigen muss. Bei teachTool stellt

also eine Einschränkung des Autors den Vorteil dar, dass dieser dem E-Learning angemessene im Design funktionale Module erstellt.
Der Zusammenhang zwischen der Visualisierung von mathematischem Lehrstoff und dem Lernerfolg von Schülern wird in dem Artikel: „Students learn better when the Numbers don't talk and dance“ [Gra05] beschrieben: „Researchers found that when college students were taught an artificial form of mathematics and physics, they learned it better when it was presented using simple, abstract symbols – such as plain stars and raindrops – rather than more visually engaging and concrete 3-D objects that moved dynamically on a computer screen.“ Als Grund für die gemachten Beobachtungen nennt Sloutsky: „... concrete objects have more 'perceptual richness', meaning there is more for students to look at and process. That means there is more to distract students from what is important.“

Kapitel 7

Mathematische Formeln

Eine Schwierigkeit bei der Erstellung von E-Learning-Modulen, gerade in der Mathematik und den Naturwissenschaften, ist die Integration von Formeln.
teachTool stellt für die Eingabe einen Formel-Editor zur Verfügung (siehe Abbildung 7.1).
Die Formeln werden mit in der Mathematik gängigen LaTeX-Befehlen eingegeben. Zur Vereinfachung gibt es jedoch einige Buttons[1], die den Code für z.B. einen Bruch oder eine Summe automatisch erstellen.
Die Eingabe wird durch HotEqn der Ruhr-Universität Bochum [Ruh] dargestellt, sodass eine permanente Vergleichsmöglichkeit zwischen eingegebenem Code und dem erzeugten Ergebnis gegeben ist. Durch dieses Vorgehen ist es auch nicht LaTeX-erfahrenen Benutzern möglich, Formeln zu erzeugen.

Ferner kann der Autor unter Windows das Programm TeXaide der Firma Design Science [Des] verwenden. Dieses kann kostenlos heruntergeladen werden und funktioniert genauso wie der Formel-Editor von Microsoft Word. Die Formeln können von Word nach TeXaide kopiert und von dort mittels Copy&Paste in den teachTool-internen Formel-Editor übertragen werden, da TeXaide sie in LaTeX-Code umwandelt. Auch schon mit Word erzeugte Formeln können auf diesem Wege importiert werden.
Es wird bei der Eingabe von Formeln also das Grundkonzept von teachTool umgesetzt, ohne Spezial-(LaTeX-)Wissen Module erstellen zu können. Gleichzeitig ist es LaTeX-erfahrenen Benutzern möglich, ihre Kompetenzen einzusetzen und (beliebig) komplexe Formeln zu erzeugen. Das ist für viele Mathematiker und Physiker von absoluter Priorität, da sie LaTeX als Standard ansehen, der nicht durch einen reinen WYSIWYG Formel-Editor ersetzbar ist.

Für die Darstellung der Formeln in der HTML-Version der Lerneinheiten werden sie in GIF-Dateien umgewandelt.

[1] Die Symbole der Buttons entstammen mit freundlicher Erlaubnis der Autoren dem LaTeX-Editor TeXnicCenter [Wie].

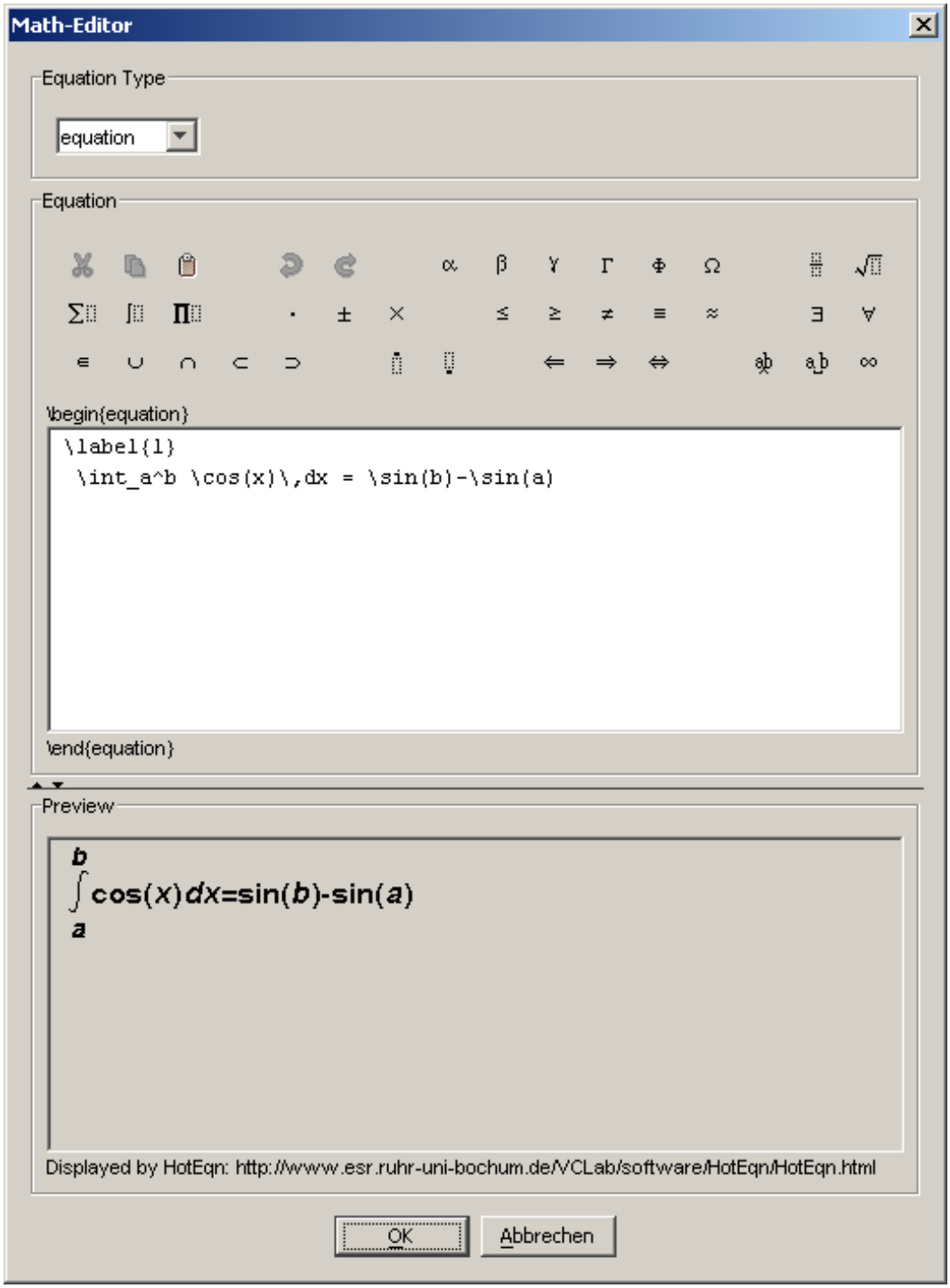


Abbildung 7.1: teachTool Formel-Editor

Der sich zur Zeit sehr stark entwickelnde Standard MathML [W3C], zum Darstellen von Formeln in HTML-Seiten, benötigt auf einigen Browsern noch Plug-Ins und ist auf älteren Browsern evtl. nicht verfügbar. Es wurde deshalb aus Kompatibilitätsgründen der Weg über die Darstellung mit Bilddateien gewählt.
Dafür werden zwei Laufzeitumgebungen von teachTool unterschieden.

- Ist auf einem Computer LaTeX installiert, wird dieses verwendet,
- sonst werden die GIFs über die paint()-Methode von HotEqn generiert.

Die über HotEqn erzeugten Formelbilder sind qualitativ etwas schlechter als die Ergebnisse, die z.B. von latex2html produziert werden. Die Installation von latex2html ist jedoch insbesondere auf Windows-Betriebssystemen recht aufwändig. Daher wurde ein Weg gefunden, nur über die Konvertierung mit LaTeX und die Java Pakete jDvi [Hof] und ACME [Pos] entsprechende GIF-Dateien zu erstellen.
Eine Herausforderung war dabei die vertikale Ausrichtung der Formelbilder im Fließtext.
Im folgenden wird am Beispiel der Formel `$\frac{1}{2}$` beschrieben, wie sie in ein GIF umgewandelt wird.

1. Es werden die folgenden LaTeX-Befehle in einer TEX-Datei gespeichert:

   ```
   \documentclass[10pt]{amsart}
   \usepackage[latin1]{inputenc}
   LATEX HEADER
   \begin{document}
   \thispagestyle{empty}
   x~\begin{math}\frac{1}{2}\end{math}
   \end{document}
   ```

   Der Zusatz `x~` wird nur bei Formeln im Fließtext verwendet und wird für die Ausrichtung der Formel im Text gebraucht.
   LATEX HEADER umfasst in jedem Fall das Einbinden der Pakete: `graphics`, `color` und `german`. Weitere LaTeX-Pakete können in teachTool über die Menüeinträge File → LaTeX Header hinzugefügt werden.

   So ist z.B. durch zusätzliches Einbinden des Pakets `chemtex` auch die Erzeugung der in Abbildung 7.2 zu sehenden chemischen Strukturformel möglich. Benötigt wird dafür lediglich der Befehl:
   `\sixring{$OH$}{Q}{Q}{Q}{Q}{$NC$}{S}{D}{S}`

2. LaTeX wird ausgeführt, um die TEX-Datei in eine DVI-Datei umzuwandeln.

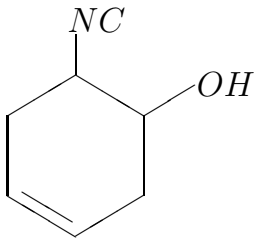
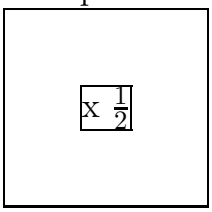


Abbildung 7.2: ChemTeX im teachTool Formel-Editor

3. Die entstandene DVI-Datei wird mit jDvi geöffnet und in ein BufferedImage gezeichnet.

4. Der Bereich des BufferedImage wird ermittelt, in dem sich die Formel befindet. D.h. es wird die erste bzw. letzte Zeile bzw. Spalte gesucht, in der sich ein nicht transparentes Pixel befindet.



5. Innerhalb des in 4. ausgewählten Bereichs, wird der Abstand des vorangestellten x zum oberen und unteren Rand berechnet.

   - Ist der Abstand zum unteren Rand 0, kann die Formel im HTML-Code durch `align = "BOTTOM"` vertikal nach unten ausgerichtet werden.
   - Ist der obere Abstand größer als der untere, wird die Formel vertikal zentriert (`align = "MIDDLE"`), der untere Abstand auf die Differenz von oberem und unterem Abstand und der obere Abstand auf 0 gesetzt. Aus technischen Gründen (s.u.) wird zum unteren Abstand 6 und zum obere Abstand 2 addiert.
   - Ist der obere Abstand nicht größer als der untere, wird die Formel vertikal zentriert (`align = "MIDDLE"`), der obere Abstand auf die Differenz von unterem und oberem Abstand und der untere Abstand auf 4 gesetzt.

   Die verwendeten Manipulationen des oberen bzw. unteren Abstands sind experimentell ermittelt worden. Sie berücksichtigen die unterschiedlichen Darstellungsweisen von Internet Explorer und Netscape bei der vertikalen Zentrierung. (Handelt es sich nicht um eine Formel im Fließtext, wird dieser Schritt übersprungen.)

6. Das BufferedImage wird auf die Größe der Formel zurecht geschnitten. Bei vorangestelltem x muss dazu noch die Spalte des BufferedImage gefunden werden, in der die Formel beginnt.

7. Das BufferedImage wird unter Verwendung der Klasse GifEncoder aus dem ACME-Paket in eine GIF-Datei gespeichert.
   Es wird nicht das PNG-Format zum Speichern des Bildes verwendet, da Internet Explorer transparente PNGs nicht korrekt darstellt. Andererseits ist es vorteilhaft, Transparenz in den Bilddateien zu verwenden, weil somit Formelbilder auch auf verschiedenen Hintergrundfarben verwendet werden können.

8. Im letzten Schritt wird analog zu latex2html in der Datei images.pl, die sich im Ordner der HTML-Dateien befindet, eine Verbindung zwischen Formel und Bilddatei hergestellt. Somit muss ein GIF zu einer Formel nur einmal erstellt werden. Bei jedem weiteren Verwenden der Formel kann die entsprechende Bilddatei über images.pl lokalisiert werden.

Kapitel 8

E-Learning-Standards

Um auch zukünftig eine Verwendung der mit teachTool erstellten Module zu gewährleisten und eine Einbettung in verschiedene Lernplattformen zu ermöglichen (siehe Abbildung 8.1), wurde eine Export-Möglichkeit programmiert, die den Standard SCORM erfüllt.

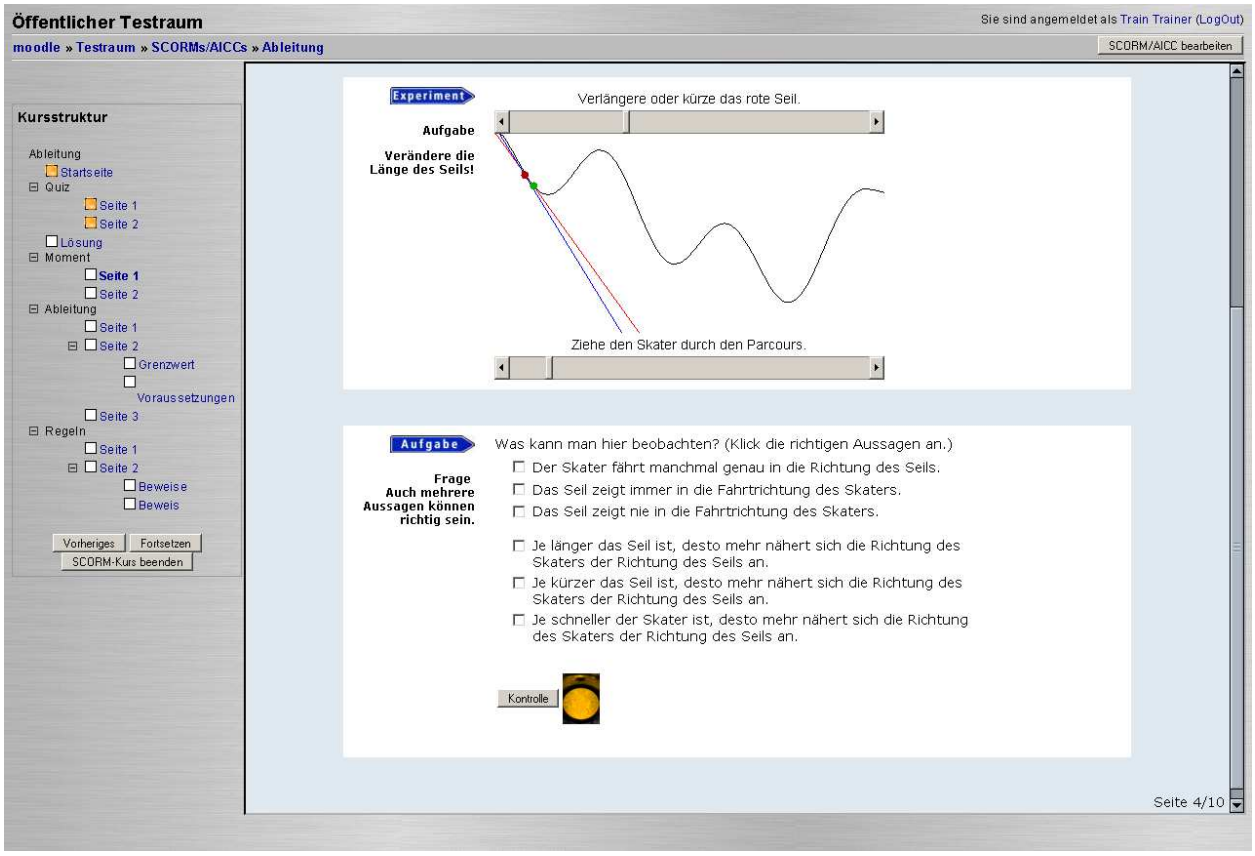


Abbildung 8.1: MathePrisma-Modul: „Ableitung“ in moodle 1.5.3+

Im Folgenden wird erläutert, wie SCORM aufgebaut ist und wie der Export in teachTool realisiert wurde.

8.1 SCORM

Die folgenden Angaben zu SCORM entstammen den Dokumentationen zu SCORM 2004 auf den Internetseiten von ADL [ADL] und einer deutschsprachigen Ausarbeitung [Glü04].
SCORM steht für Sharable Content Object Reference Model und stellt eine Sammlung von Spezifikationen und Standards dar, um Lösungen zu Problemen wie

- Zugänglichkeit,
- Anpassungsfähigkeit,
- Erschwinglichkeit,
- Beständigkeit,
- Interoperabilität
- und Wiederverwendbarkeit

für web-basiertes E-Learning zu finden.

SCORM wurde durch die Advanced Distributed Learning-Initiative (ADL) für das US-Militär entwickelt und im Jahr 2000 in seiner ersten Version als Standard veröffentlicht. Es fasst die folgenden Spezifikationen zusammen:

- IEEE Data Model
  Datenmodell für die Kommunikation zwischen Inhaltsobjekten und einem Learning Management System (LMS)
- IEEE ECMAScript Application Programming Interface
  Kommunikationsschnittstelle zwischen Inhaltsobjekten und LMS
- IEEE Learning Object Metadata (LOM)
  Metadatensatz für die Auszeichnung der Lernobjekte
- IMS Content Packaging
  Definition für die Speicherung austauschbarer Lerneinheiten
- IMS Simple Sequencing
  Regeln für den Ablauf von Lerneinheiten

SCORM gliedert sich in vier Bereiche, die Bücher genannt werden:

1. **Overview**
   Das erste Buch liefert Hintergrundinformationen und einen Überblick zu SCORM.

2. **Content Aggregation Model (CAM)**
   Im zweiten Buch werden die zwei Klassen von Lernobjekten Sharable Content Object (SCO) und Asset definiert und beschrieben. Aufbauend auf diesen Grundobjekten wird festgelegt, wie sie zu Lerneinheiten zusammengefasst werden, wie die Auszeichnung durch Metadaten erfolgt und wie die Lerneinheiten mittels Content Packaging zum Austausch mit anderen Systemen abgespeichert werden müssen.

3. **Run-Time Environment (RTE)**
   Dieses Buch definiert das Laden von SCOs und Assets durch ein LMS. Es werden außerdem Regeln für die Kommunikation zwischen SCOs und LMS und Formate für die Speicherung von Daten der SCOs festgelegt.

4. **Sequencing and Navigation (SN)**
   Das vierte Buch regelt die Ablaufsteuerung und Navigation. Es können sowohl durch den Lernenden als auch durch das System Ereignisse ausgelöst werden, die den Ablauf einer Lerneinheit beeinflussen.

8.1.1 Content Aggregation Model

Die Inhaltsorganisation findet in SCORM (nach den Vorschriften von Content Packaging) in der Datei imsmanifest.xml in XML-Syntax statt. In dieser wird eine hierarchische Organisation der Inhalte vorgenommen, aus denen die Lerneinheit besteht. Physikalische Dateien können durch SCOs bzw. Assets als Ressourcen gebündelt und in die Inhaltsorganisation eingebracht werden (siehe Abbildung 8.2).

SCOs bieten im Gegensatz zu Assets die Möglichkeit, mit dem Learning Management System (LMS) kommunizieren zu können. So ist es z.B. möglich HTML-Seiten als SCOs einzubinden, die über JavaScript mit dem LMS Daten über die Bearbeitung von Interaktionen austauschen. Die HTML-Seiten besitzen z.B. wiederum Bilder, die als Assets als statische Ressourcen eingebunden werden.

Ebenfalls über die Datei imsmanifest.xml werden Metadaten zu Lernobjekten bzw. Lerneinheiten hinzugefügt. Die Metadaten gliedern sich basierend auf LOM in folgende neun Kategorien:

1. General

2. Lifecycle

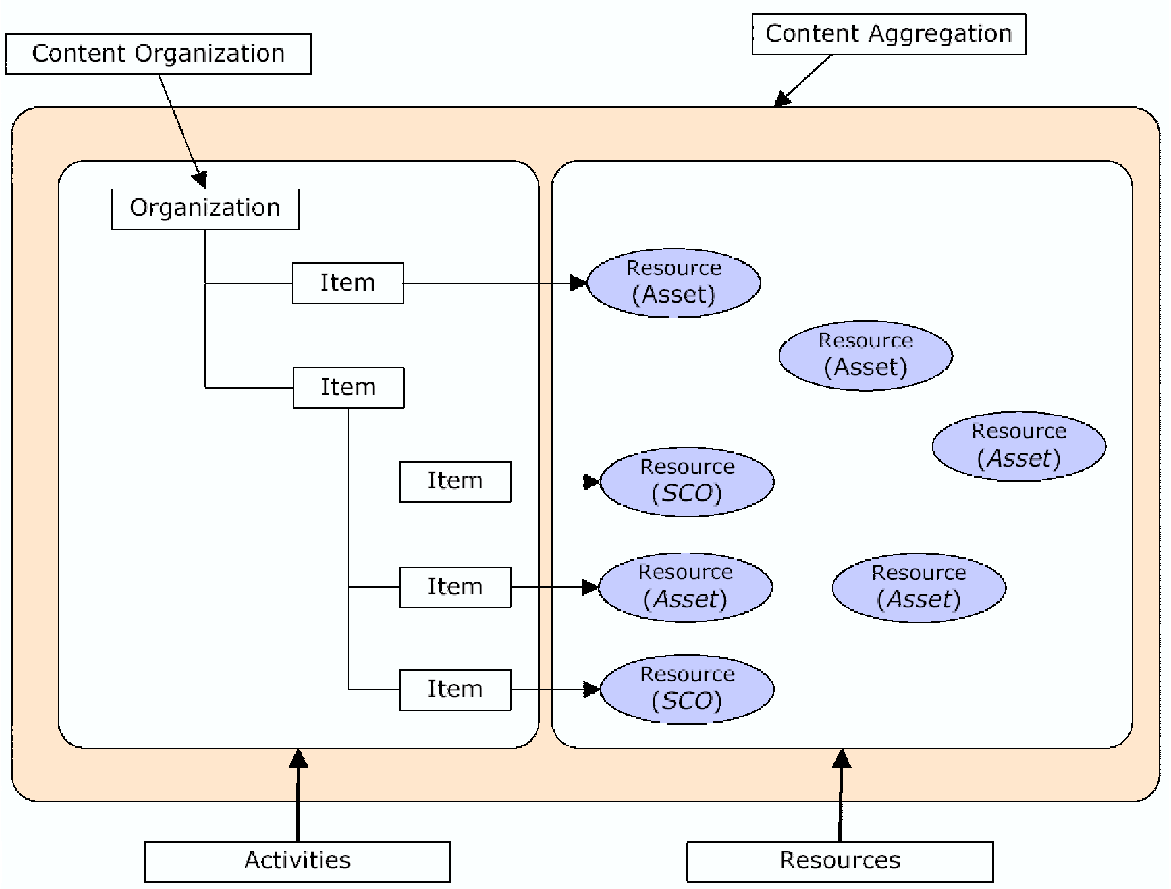


Abbildung 8.2: SCORM Content Organization; Quelle: Advanced Distributed Learning (ADL), Sharable Content Object Reference Model (SCORM®) Content Aggregation Model Version 1.3.1, 2004.

3. Meta-Metadaten
4. Technical
5. Educational
6. Rights
7. Relation
8. Annotation
9. Classification

Die Speicherung der Lerneinheiten findet nach Content Packaging in einem Package Interchange File (PIF) statt, das sehr oft als ZIP-Archiv realisiert wird. In diesem

PIF befindet sich die Datei imsmanifest.xml im Wurzelverzeichnis. Alle benötigten Dateien werden, gemäß der in imsmanifest.xml gemachten Ordnerangaben, gespeichert.

8.1.2 Run-Time Environment

Die Kommunikation und der Datenaustausch zwischen LMS und SCO findet über das Application Programming Interface (API) statt (siehe Abbildung 8.3).

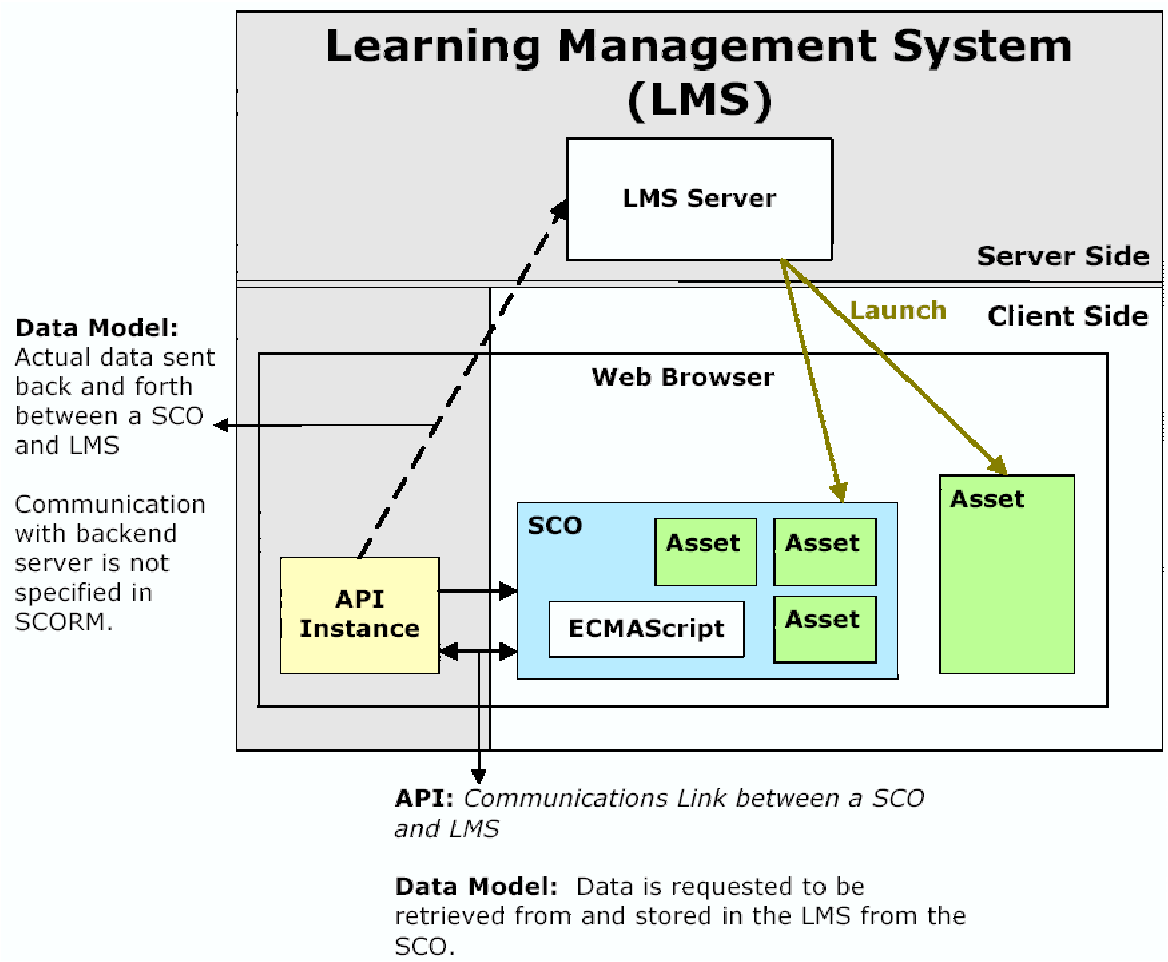


Abbildung 8.3: SCORM Run-Time Environment; Quelle: Advanced Distributed Learning (ADL), Sharable Content Object Reference Model (SCORM®) Run-Time Environment Version 1.3.1, 2004.

Nachdem das LMS ein SCO gestartet hat, muss dieses die API finden, um einen Datenaustausch aufbauen zu können. Dazu wird in SCORM festgelegt, wo sich die

API, die oft als Applet realisiert wird, befinden darf.
Für die Kommunikation zwischen LMS und SCO gibt es 8 API-Methoden:

- **Session Methoden**

  1. return_value = Initialize(parameter)
  2. return_value = Terminate(parameter)

- **Datentransfer Methoden**

  3. return_value = GetValue(parameter)
  4. return_value = SetValue(parameter1, parameter2)
  5. return_value = Commit(parameter)

- **Support Methoden**

  6. return_value = GetLastError()
  7. return_value = GetErrorString(parameter)
  8. return_value = GetDiagnostic(parameter)

Jedes SCO muss, nach dem Finden der API, mindestens die beiden Session Methoden aufrufen. Initialize("") leitet die Kommunikation ein, erst dann ist ein Datenaustausch über die Datentransfer Methoden möglich. Bevor ein neues SCO vom LMS gestartet wird, muss das aktuelle SCO Terminate("") aufrufen.
Der Datenaustausch wird über die Datentransfer Methoden und ein vorgeschriebenes Datenmodell realisiert. Das Datenmodell berücksichtigt das Speichern folgender Informationen:

- Informationen über den Lernenden
- Interaktionen, die der Lernende mit dem SCO hatte
- Lernzielinformationen
- Erfolgsstatus
- Beendigungsstatus

Die drei Support Methoden dienen der Fehlerbehandlung.

8.1.3 Sequencing and Navigation

Das vierte SCORM-Buch beschreibt Möglichkeiten, wie der Ablauf einer Lerneinheit beeinflusst werden kann. Es kann hierüber dem Lernenden z.B. untersagt werden bestimmte Aktivitäten zu starten bevor dafür erforderliche Aktivitäten bearbeitet wurden.
Wie in Kapitel 4 beschrieben besitzen mit teachTool erstellte Lerneinheiten eine flache, zumeist lineare Struktur. Ferner ist es nach Meinung des Autors nicht wünschenswert, den Lernenden in seiner Auswahl zu beschneiden. Daher wurden in teachTool beim SCORM Export keine speziellen Ablaufregeln festgelegt, weshalb auf eine genauere Beschreibung dieses Buchs verzichtet wird.

8.2 SCORM Export in teachTool

In teachTool gibt es unter dem Menüpunkt „Extras“ den Eintrag „SCORM Export“. Über diesen kann zu einer Lerneinheit ein SCORM-konformes ZIP-Archiv erstellt werden.
Die generierten Content Packages wurden mit den Lernplattformen ILIAS [ILI] in Version 3.4.0, moodle [moo] in Version 1.5.3+ und dem Reload SCORM 1.2 Player [REL] in Version 1.2.1 auf Standardkonformität getestet. Sie erfüllen die Anforderungen von SCORM 1.2.

Die Inhaltsorganisation in der Datei imsmanifest.xml findet analog zur Erstellung der Lerneinheiten hierarchisch statt. Es schachteln sich die „items“ für Kapitel, (Hauptpfad-)Seite und Nebenpfadseite ineinander. Somit zeigt ein LMS, das eine Lerneinheit präsentiert, einen ähnlichen Navigationsbaum, wie teachTool bei der Bearbeitung.
Es folgt ein Auszug aus dem XML-Code der Datei imsmanifest.xml:

```
<organizations default="ORG-MODULE">
  <organization identifier="ORG-MODULE"
                structure="hierarchical">
    <title>Modulname</title>
    <item identifier="ITEM-STARTPAGE-NODE1" isvisible="true"
          identifierref="RES-NODE1">
      <title>Startseite</title>
    </item>
    ...
    <item identifier="ITEM-CHAPTER-NODE8" isvisible="true">
      <title>Viertes Kapitel</title>
      <item identifier="ITEM-PAGE-NODE8" isvisible="true"
```

```
            identifierref="RES-NODE8">
          <title>Seite 1</title>
          <item identifier="ITEM-SUBPAGE-NODE9"
                isvisible="true" identifierref="RES-NODE9">
            <title>Verweis</title>
          </item>
        </item>
        <item identifier="ITEM-PAGE-NODE10" isvisible="true"
              identifierref="RES-NODE10">
          <title>Seite 2</title>
        </item>
      </item>
      ...
    </organization>
  </organizations>
  <resources>
    <resource identifier="RES-NODE1" type="webcontent"
              adlcp:scormtype="sco" href="pages/node1.htm">
      <file href="pages/node1.htm" />
      <dependency identifierref="RES-ALL-ASSETS" />
    </resource>
    ...
  </resources>
```

Es werden außerdem in imsmanifest.xml folgende Metadaten über die Lerneinheiten gespeichert:

- General → Title: Name des Moduls
- General → Description: Untertitel des Moduls
- Lifecycle → Contribute → Centity: Namen der Autoren
- Lifecycle → Contribute → Date: Monat der Erstellung
- Educational → Context: Zielgruppe

Für die JavaScript-Implementierung der Kommunikation zwischen SCO und API werden die Dateien APIWrapper.js und SCOFunctions.js des Reload Editors [REL] in Version 2.0.2 verwendet. Diese implementieren die 8 API-Methoden aus den Bereichen Session, Datentransfer und Support.

Der Aufruf von Initialize() wird in jedem SCO über die loadPage() Funktion aus SCOFunctions.js realisiert. Die API-Methode Terminate() ist Bestandteil der

JavaScript-Funktion unloadPage(status) aus SCOFunctions.js und wird in den SCOs auf das Event „onUnload“ hin ausgeführt. Der Parameter 'status' wird dabei stets auf 'incomplete' gesetzt, damit das SCO auch später noch angezeigt werden kann. Der Wert 'completed' führte dazu, dass ein SCO beim nächsten Aufruf vom LMS geblockt würde.
Außerdem wird über loadPage() und unloadPage(status) eine Zeitmessung ausgeführt und über den Eintrag 'cmi.core.session_time' im Datenmodell gespeichert. Es kann so für jedes SCO überprüft werden, wie lange ein Lernender es bearbeitet hat.

Ein weiterer Datenaustausch zwischen LMS und SCO findet bei verschiedenen Interaktionsformen statt. Für die Aufgabentypen

- Antwortfeld,
- Multiple Choice,
- Hotspot
- und Applet

können vom Autor in teachTool Punkte vergeben werden. Wird eine Aufgabe in einem SCO vom Lernenden bearbeitet und der Auswertungsbutton gedrückt, sendet das SCO mittels SetValue() Daten über die Antwort des Lernenden und die erreichte Punktzahl an das LMS. Bei jedem weiteren Bearbeiten des SCO wird von jedem dieser Aufgabentypen mit GetValue() überprüft, ob die Aufgabe schon korrekt bearbeitet wurde. Wenn das der Fall ist, wird z.B. in ein Antwortfeld schon der richtige Text eingetragen.

Die folgende Abbildung 8.4 zeigt, am Beispiel von moodle, wie die von der Lernplattform gespeicherten Lern-Daten (dem Lehrenden) zum Analysieren präsentiert werden können:

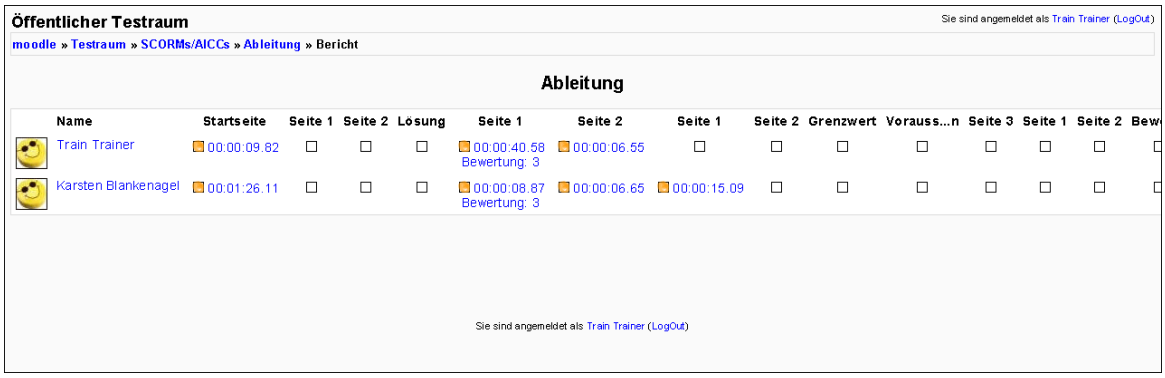


Abbildung 8.4: MathePrisma-Modul: „Ableitung“ Bericht in moodle 1.5.3+

Kapitel 9

Software-Ergonomie

Bei der Implementierung von Anwender-Software spielt der Aspekt der Verwendbarkeit (Usability) und Erlernbarkeit durch den Anwender eine wesentliche Rolle. Grundlegende Regeln, die bei der Programmierung graphischer Benutzeroberflächen zu beachten sind, werden in der Software-Ergonomie festgelegt.
„Ziel der Software-Ergonomie ist die Entwicklung und Evaluierung gebrauchstauglicher Software-Produkte, die Benutzer zur Erreichung ihrer Arbeitsergebnisse befähigen und dabei ihre Belange im jeweiligen Nutzungskontext beachten (in Anlehnung an /EN ISO 9241-11: 1998/).“ (siehe [Bal00], Seite 485)
Autorensysteme sollen das Erstellen von Lerneinheiten ohne Kenntnis von Programmiersprachen ermöglichen. Zum Kreis der Anwender gehören in erster Linie Lehrende an Schulen oder Universitäten und im Fall von teachTool insbesondere zusätzlich Lehramtsstudierende. Des Weiteren werden bei teachTool auch Schüler zum Anwenderkreis gezählt. Diese arbeiteten z.B. im Rahmen von Schülerpraktika erfolgreich mit teachTool. Die Berücksichtigung von Schülern als Autoren wird auch durch Heyder [Hey00] unterstrichen, der die Einsatzmöglichkeiten von Autorensystemen in der Schule evaluiert hat.
„Auf den Einsatz von Autorensystemen im Unterricht bezogen ist insbesondere die Benutzerfreundlichkeit für die Schüler ein sehr wichtiges Kriterium, denn sie sollen nicht erst eine Programmiersprache mühselig erlernen müssen, sondern möglichst ohne große Vorkenntnisse mit dem Autorensystem umgehen können.“ (siehe [Hey00], Seite 128)
Neben der Einhaltung der software-ergonomischen Kriterien erweist es sich als vorteilhaft, sich an gängigen Softwareprodukten zu orientieren, die von der zu erwartenden Autorengruppe häufig verwendet werden. Hier bieten sich insbesondere Microsoft-Produkte wie Word oder Excel an, mit deren Bedienweisen viele PC-Benutzer vertraut sind.
„Grafikbildschirme und grafische Benutzeroberflächen für Windows-Systeme und Web-Browser werden als Standard angesehen.“ (siehe [Bal00], Seite 487)
Natürlich stehen Usability-Kriterien und „Bedienstandards“ gängiger Programme

selten im Widerspruch zueinander. Vielmehr sind sie zumeist voneinander abgeleitet.

Bei der Entwicklung von teachTool sind vor allem fünf Überlegungen zu nennen, mit denen software-ergonomische Kriterien berücksichtigt wurden:

- Grundaufbau des Hauptfensters ähnlich dem Windows Explorer
- WYSIWYG („What You See Is What You Get“)
- Gruppierung zusammengehöriger Komponenten
- Verwendung von Assistenten (Wizards)
- Bereitstellung von Buttons in Menü- und Symbolleisten

Um deutlich zu machen, wie die genannten Kriterien in der aktuellen Version des Programms im Gegensatz zu einer älteren Version berücksichtigt wurden, finden sich in Abbildung 9.1 und 9.2 Screenshots der Oberfläche (die ältere Version trägt noch den Arbeitstitel „MathePrisma-Editor“).

9.1 Grundaufbau des Hauptfensters ähnlich dem Windows Explorer

Schon früh wurde entschieden, die Benutzeroberfläche in ähnlicher Weise zu gestalten, wie sie beim Windows Explorer vorzufinden ist. Gründe dafür sind die leichtere Orientierung für Autoren auf der teachTool-Oberfläche, die mit Windows Explorer vertraut sind und die ähnliche Funktionalität der beiden Programme. Es gibt also auf der linken Seite einen Objekt-Baum, der die Struktur der zu bearbeitenden Lerneinheit widerspiegelt. Wird ein Knoten im Baum angeklickt, erscheint im rechten Fensterbereich die Information, die dem ausgewählten Knoten zugeordnet ist. Der Baum verdeutlicht zum einen den hierarchischen Aufbau des Lernmoduls. Zum anderen ermöglicht er per Ziehen und Ablegen (Drag & Drop) die Umstrukturierung der Lerninhalte. Den Vorteil dieser Technik beschreibt Balzert wie folgt: „Die Zwischenablage stellt einen Umweg beim Datenaustausch dar. Durch 'Kopieren' bzw. 'Ausschneiden' 'verschwinden' die Daten in einer für den Benutzer unsichtbaren Senke und werden mittels 'Einfügen' in der Zielanwendung wieder sichtbar. Neuere Anwendungen unterstützen daher zusätzlich das direkte 'Ziehen und Ablegen' von Objekten zwischen Anwendungen.“ (siehe [Bal00], Seite 508)

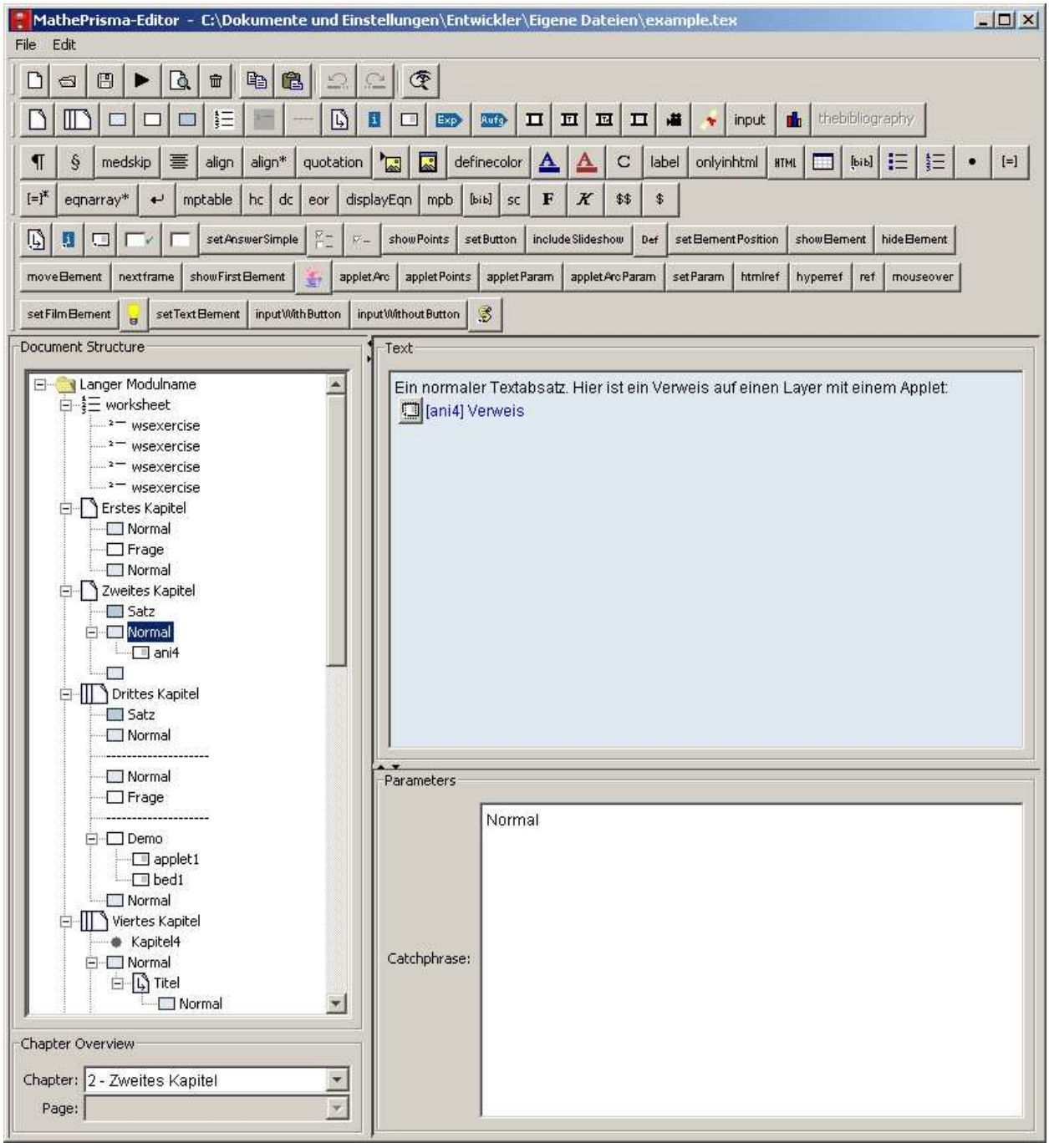


Abbildung 9.1: Oberfläche einer älteren teachTool Version

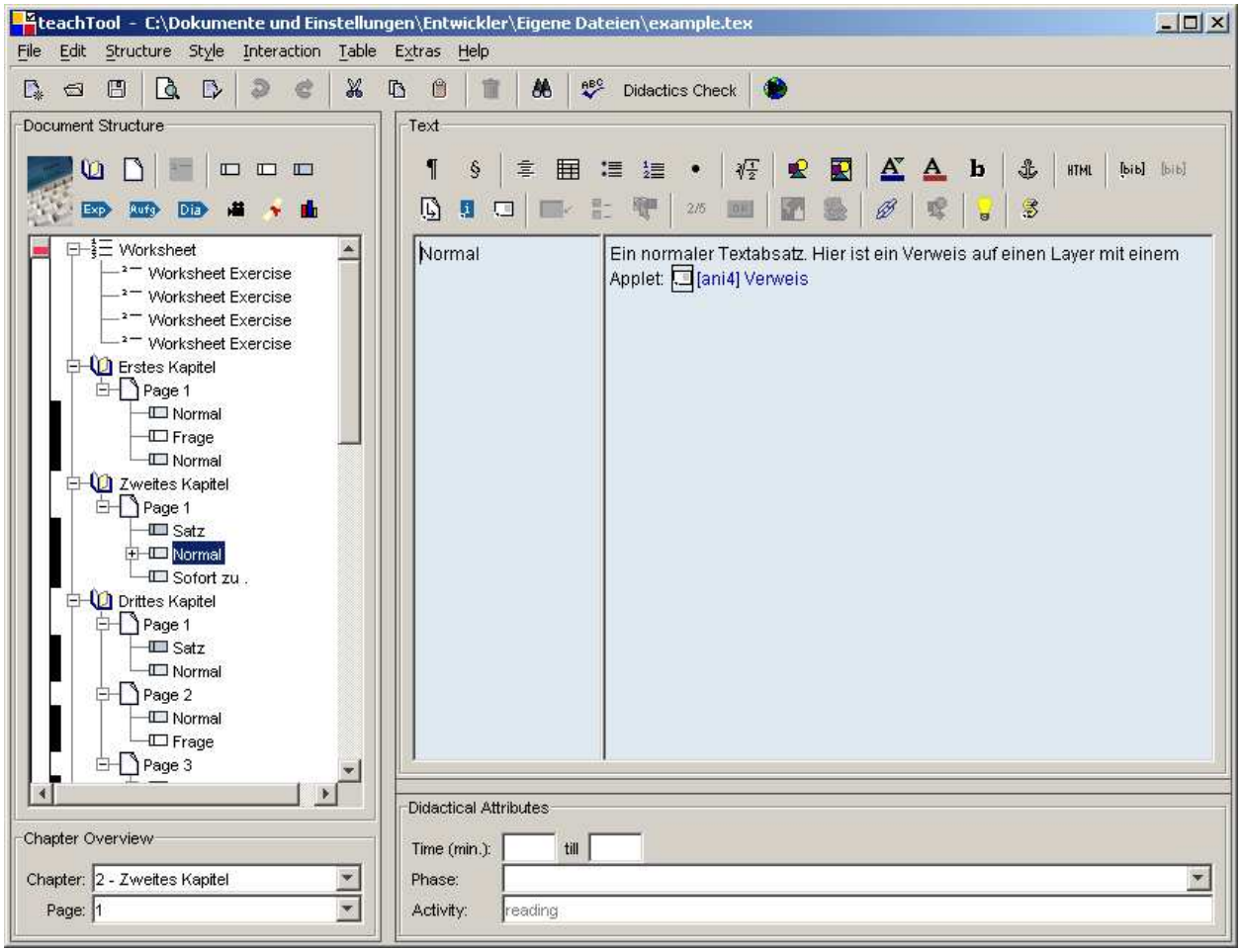


Abbildung 9.2: Oberfläche der aktuellen teachTool Version

9.2 WYSIWYG

Im Screenshot der älteren Version der Oberfläche befindet sich unten rechts ein Textfeld mit der Beschriftung „Catchphrase“. Dieses enthält den Text, der den im Strukturbaum ausgewählten Abschnitt in kurzen Worten beschreibt. Da dieser Text in der HTML-Variante der Lerneinheit jeweils in einer Spalte links vom jeweiligen Abschnitt steht und dieselbe Hintergrundfarbe wie dieser besitzt, wurde das entsprechende Textfeld in der neuen Version links neben das Abschnittstextfeld gesetzt. Diese Veränderung erlaubt dem Autor einer Lerneinheit ein schnelleres Wechseln zwischen den Ebenen der Bearbeitungs- und der HTML-Ansicht.
Leider kann bei der Bearbeitungs-Ansicht nicht immer dieselbe Darstellung wie in der HTML-Ansicht verwendet werden. So kann z.B. ein Link bei der Eingabe nicht einfach nur als unterstrichener Text angezeigt werden, da er verschiedene Ziele besitzen kann (z.B. Nebenpfadseite oder Infoseite). Es wurde jedoch darauf geachtet, die Unterschiede so gering wie möglich zu halten.

9.3 Gruppierung zusammengehöriger Komponenten

Ein weiterer Unterschied zwischen alter und neuer Version besteht in der Positionierung der Symbolleisten. Schon in der alten Version gibt es vier verschiedene Symbolleisten, die von oben nach unten folgende Funktionen gruppieren:

- Datei- bzw. Bearbeiten-Funktionen
- Hinzufügen von Strukturobjekten in den Baum
- Stilelemente für die Textgestaltung
- Interaktionen im Textbereich

Diese vier Leisten sind für einen Benutzer des Programms, der sich mit dessen Funktionalitäten nicht oder wenig auskennt, verwirrend und wenig intuitiv. Zum einen muss man bei der Vielzahl an Buttons lange suchen bis man den richtigen findet. Zum anderen ist nicht klar, in welchem Bereich des Hauptfensters die jeweiligen Buttongruppen Verwendung finden.
Aus diesen Gründen wurde die Anzahl der Buttons reduziert. Außerdem wurden die zweite Symbolleiste dem Strukturbaum und die dritte und vierte Symbolleiste dem Textbereich rechts zugeordnet. Dies entspricht dem bei Balzert empfohlenen Vorgehen: „Mehrere Elemente können durch räumliche Nähe, räumliche Anordnung, [...] zu einer Gruppe oder Makroeinheit zusammengefasst werden.“ (siehe [Bal00], Seite 616)

9.4 Verwendung von Assistenten (Wizards)

Beim Erstellen einer Diashow gibt es bei teachTool einige Freiheitsgrade. Es muss festgelegt werden:

- Welche Art von Diashow eingesetzt werden soll.
- Welche Grundelemente (Bilder, Texte, ...) für die Dias zur Verfügung stehen sollen.
- Wie die Dias aus den einzelnen Grundelementen zusammengestellt werden sollen.

In der älteren Programmversion musste der Anwender in den Strukturbaum den ausgewählten Diashowtyp einfügen und sich dann in einem recht unübersichtlichen Verfahren Grundelemente und Diashowablauf zusammenklicken.
Da Diashows ein sehr oft verwendetes Interaktionselement darstellen, wurde in der

neuen Version die Komposition von Diashows über einen Assistenten realisiert. Der Anwender muss in den Strukturbaum lediglich einen variabel verwendbaren Diashow-Eintrag einfügen. Der Rest der Eingabe findet in drei Dialogfenstern (siehe Abbildung 9.3) statt, die nacheinander durchlaufen werden. Dieser Assistent erlaubt das Zusammenstellen einer „einfachen“ Diashow mit weinigen Klicken. Er ermöglicht andererseits aber auch die Gestaltung einer an individuelle Vorstellungen angepassten Variante.
Die Technik eines Assistenten ist vielen Anwendern vertraut. Sie findet z.B. Verwendung beim Diagramm-Assistenten von Microsoft Excel.

9.5 Bereitstellung von Buttons in Menü- und Symbolleisten

Die alte teachTool Oberfläche zeigt im Menü nur die Einträge „File“ und „Edit“. Zu den Buttons der zweiten, dritten und vierten Symbolleiste befinden sich dort also keine Einträge. Balzert beschreibt den Vorteil von Menüs folgendermaßen: „Menüs sind für Anfänger und Gelegenheitsbenutzer gut geeignet. Experten werden durch Menüs dagegen in ihrem Arbeitsfluss oft 'gebremst'.“ (siehe [Bal00], Seite 525) In der neuen Version gibt es zu jeder Symbolleiste einen gesonderten Eintrag im Menü. Das hebt zum einen die unterschiedliche Bedeutung der verschiedenen Symbolleisten hervor. Zum anderen vereinfacht gerade das gleichzeitige Anzeigen von Symbolen und Funktionstexten in den Menüs die Einarbeitung in das Programm.

9.6 Weitere software-ergonomische Merkmale von teachTool

Zusätzlich zu den oben ausführlich beschriebenen fünf Merkmalen zur Erreichung einer hohen Benutzerfreundlichkeit von teachTool wurden die folgenden vier bei der Implementierung umgesetzt:

- **Hilfefunktion**
  teachTool besitzt eine Online-Hilfe, welche den Aufbau der Oberfläche, die ersten Schritte zum Anlegen eines neuen Moduls, die Menüs Struktur, Format und Interaktionen, die Interaktionsformen Diashow, Hotspot, Puzzle und MouseOver und das Erstellen von Formeln erklärt. Die Hilfe zur Diashow kann zusätzlich kontextsensitiv über den Assistenten zum Erzeugen von Diashows aufgerufen werden.

- **Fortschrittsanzeigen**
  Bei länger laufenden Prozessen wie z.B. dem Umwandeln nach HTML oder PDF werden Fortschrittsanzeigen eingeblendet, die den Benutzer über die im Hintergrund ablaufenden Prozesse informieren.

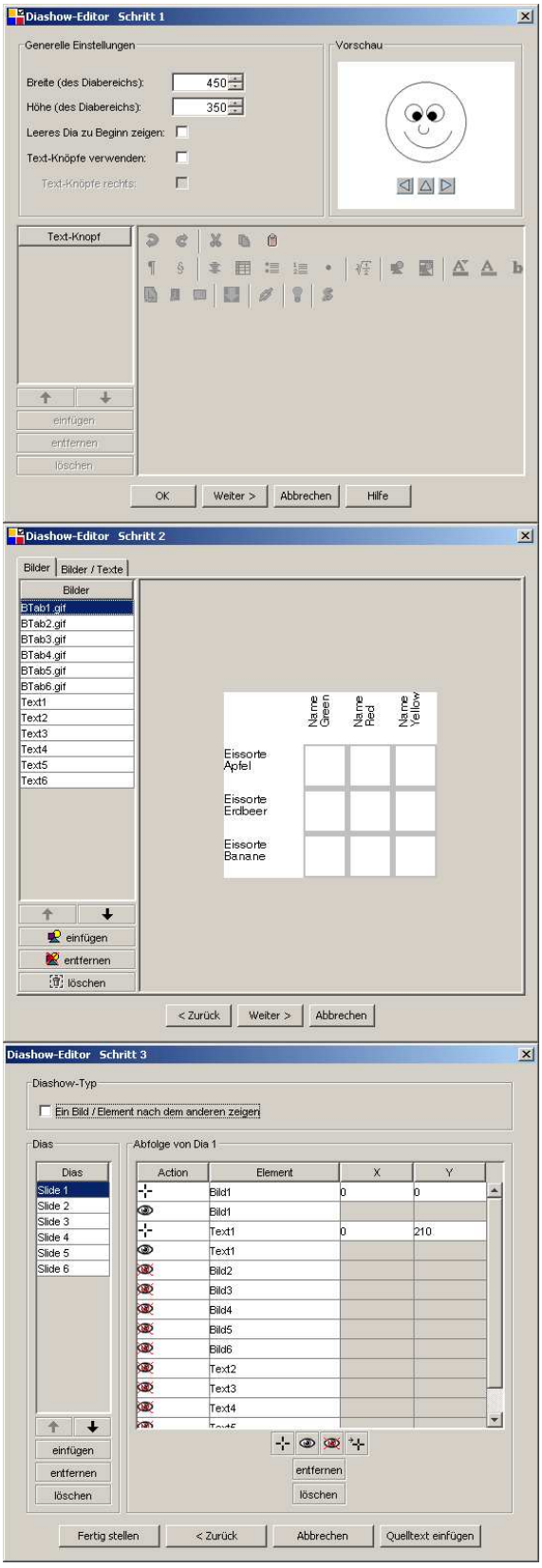


Abbildung 9.3: Diashow Dialogfenster

- **Symbole und Icons**
  In den Menü- und Symbolleisten werden – wenn möglich – Graphiken aus dem Sun „Java look and feel Graphics Repository“[1] verwendet. Ihre Bedeutung kann intuitiv verstanden werden, und sie sehen den vielen Benutzern von Microsoft-Produkten vertrauten Symbolen sehr ähnlich.

- **Reihenfolge in der Menüleiste**
  Für die Reihenfolge der Untermenüs in der Menüleiste wurde die von vielen Programmen gewohnte Anordnung Datei, Bearbeiten, [...], Extras, Hilfe gewählt.

[1] http://java.sun.com/developer/techDocs/hi/repository/

Kapitel 10

Software-Design und Erweiterbarkeit

teachTool wurde als Java-Swing-Applikation implementiert und benötigt eine Java Virtual Machine ab Version 1.4.2[1]. In diesem Kapitel werden die Bedeutung und die Zusammenhänge der Java-Klassen von teachTool beschrieben. Außerdem wird gezeigt, wie der Quelltext von teachTool durch einen Programmierer zum Einfügen neuer Grundelemente erweitert werden kann.

10.1 Das Klassen-Konzept von teachTool

teachTool besteht aus 59 Java-Klassen, die zumeist sehr eng über die Hauptklasse *TeachTool* miteinander verbunden sind. Bei den 59 Klassen sind nicht die Klassen aus den verwendeten Paketen HotEqn [Ruh], jDvi [Hof] und ACME [Pos] und Unterklassen wie z.B. verschiedene Listener berücksichtigt.
Das Zusammenwirken der Klassen kann in der Javadoc-Dokumentation zu teachTool nachgeschlagen werden. Ferner gibt es auf Seite 78 ein UML-Diagramm, das Referenzierungsmöglichkeiten von *TeachTool* zu Objekten, insbesondere der GUI-Klassen, graphisch darstellt.

Die Klasse *TeachTool* verwaltet globale Variablen, die zum Teil über Optionenfenster, wie z.B. Designoptionen, im laufenden Programm verändert werden können. Außerdem wird in ihr das Layout der graphischen Benutzeroberfläche (GUI) festgelegt. Auch die verschieden GUI Komponenten sind über globale Variablen in *TeachTool* erreichbar.

[1] In Java Virtual Machines der Versionen 1.4.0 und 1.4.1 gab es Probleme bei den Berechnungen in getPreferredSize() für Komponenten, die in ein JTextPane eingefügt wurden bzw. beim Kopieren und Einfügen von Texten unter Linux.

10.1.1 GUI

Die GUI wird aus den folgenden 13 Klassen gebildet, die wie in Abbildung 10.1 dargestellt ist, angeordnet sind:

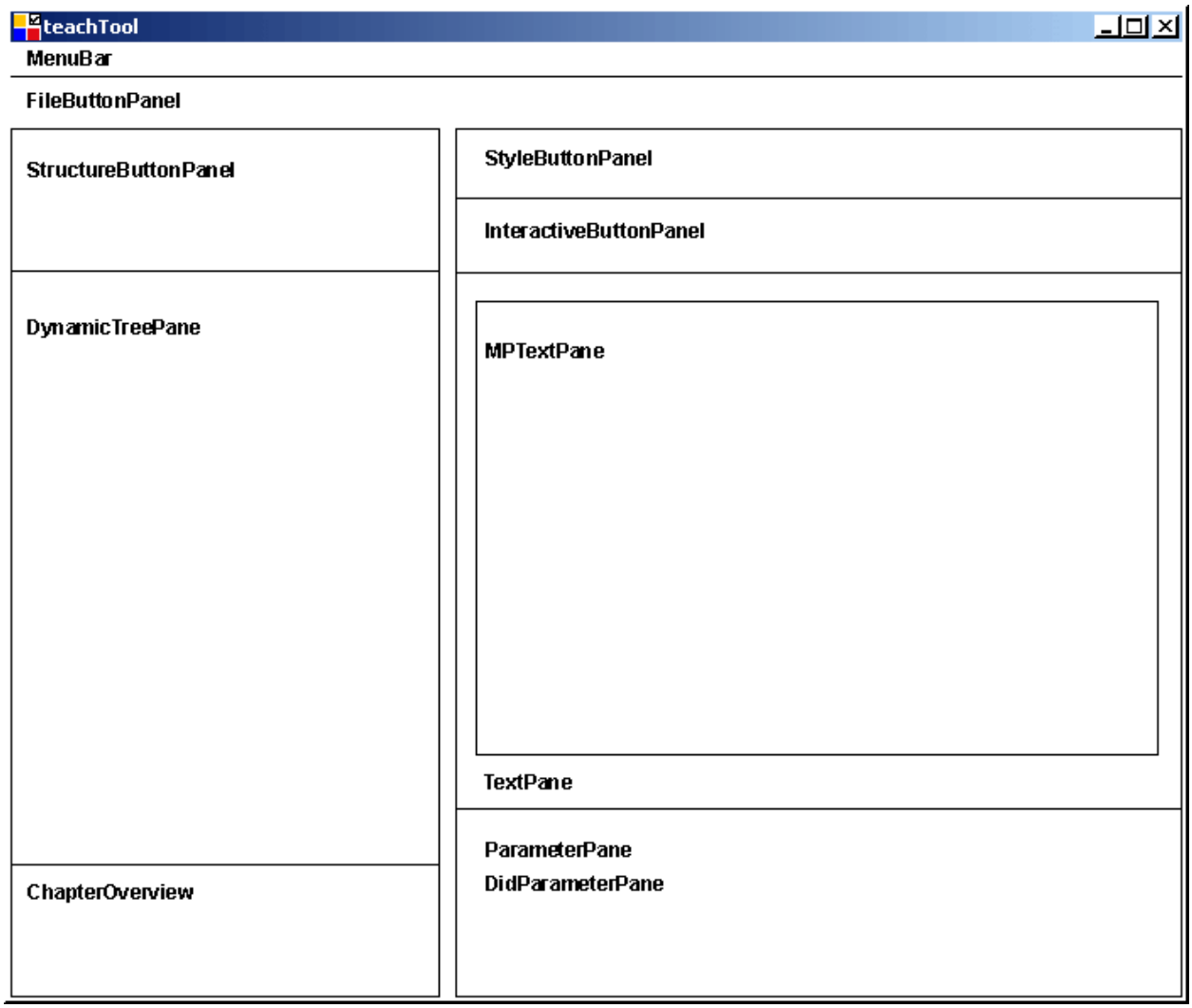


Abbildung 10.1: Die Java-Klassen der teachTool Oberfläche

- *MenuBar*
  Im Menü werden die Einträge File, Edit, Structure, Style, Interaction, Table, Extras und Help angeboten. Die meisten Aktionen, die durch die Menüeintrage ausgelöst werden, sind in den entsprechenden Symbolleisten definiert. So bieten die Untermenüs File und Edit fast die gleichen Knöpfe wie *FileButtonPanel*. Ebensolche Zusammenhänge gibt es zwischen Structure und *StructureButtonPanel*, Style und *StyleButtonPanel* und Interaction und *InteractiveButtonPanel*.

- *FileButtonPanel*
  In dieser Symbolleiste werden übergeordnete Funktionen zur Verfügung gestellt, wie Öffnen und Speichern von Modulen, Bearbeiten, Suchen, die Rechtschreibprüfung, „Didactics Check“ und das Übersetzen nach HTML.

- *StructureButtonPanel*
  StructureButtonPanel enthält Buttons zum Einfügen von Strukturelementen wie Kapitel und Seite in den Strukturbaum.

- *DynamicTreePane*
  Diese Klasse stellt die Struktur der Lerneinheiten in Form eines Baums dar. Sie verwendet *DynamicTree*, welche das Zeichnen des Baums und die Behandlung von Ereignissen, wie wählen eines anderen Knotens, Löschen und Einfügen sowie Drag&Drop im Baum übernimmt. *DynamicTree* basiert auf einer Klasse von Richard Stanford, die den Tutorials von Sun [Sun] entnommen wurde.
  Die Unterklassen *PhaseColorPane* und *TimeLinePane* stellen die Farbenleiste zu den Lernphasen der Abschnitte und die Zeitleiste dar.

- *ChapterOverview*
  Es wird gezeigt, in welchem Kapitel und auf welcher Seite sich der momentan im Baum ausgewählte Knoten / Abschnitt befindet. Über Auswahlboxen kann auch direkt zu einem Kapitel oder zu einer Seite im Baum gesprungen werden.

- *StyleButtonPanel*
  Diese Symbolleiste ermöglicht das Formatieren von Texten und das Einfügen von Formeln, Bildern und HTML-Quelltext.

- *InteractiveButtonPanel*
  Eine Symbolleiste für das Hinzufügen von interaktiven Elementen, wie Verweise und Aufgabenformate.

- *TextPane*
  In *TextPane* stellen *MPTextPane*s den Text zum ausgewählten Knoten im Baum dar. Ist ein Textabschnitt ausgewählt, werden zwei *MPTextPane*s für eine Kurzbeschreibung des Abschnitts und den eigentlichen Text nebeneinander angezeigt, ähnlich zu den beiden Spalten auf den dazugehörigen HTML-Seiten.

- *ParameterPane*
  ParameterPane zeigt Attribute zum ausgewählten Knoten an, z.B. den Namen eines Kapitels.

- *MPTextPane / MPTextField*
  Zwei Klassen zur Darstellung der Inhalte. *MPTextPane* ist von JTextPane abgeleitet und kann, im Gegensatz zu *MPTextField*, Java-Komponenten im

Fließtext beinhalten. Auf diese Art können z.B. Tabellen und Bilder dargestellt werden. Diese Klasse unterstützt also sehr den Effekt des WYSIWYG.

- *DidParameterPane*
  Die Klasse ist eingebettet in *ParameterPane* und zeigt, falls vorhanden, didaktische Attribute des momentan bearbeiteten Knotens an.

Abbildung 10.2 zeigt ein UML-Diagramm der Klassen, von denen Objekte über die Hauptklasse *TeachTool* referenzierbar sind. Die grau eingefärbten Klassen sind 10 der 13 GUI-Klassen. Nicht aufgeführt sind *DynamicTree* und *DidParameterPane*, die über *DynamicTreePane* bzw. *ParameterPane* erreicht werden können. chosenTextPane zeigt auf das aktuell gewählte *MPTextPane*, das über die Symbolleisten verändert werden kann. Objekte der Klasse *MPTextField* werden z.B. in *ParameterPane* verwendet. Die vier Klassen in der letzten Zeile der Abbildung werden im weiteren Verlauf dieses Kapitels erläutert.

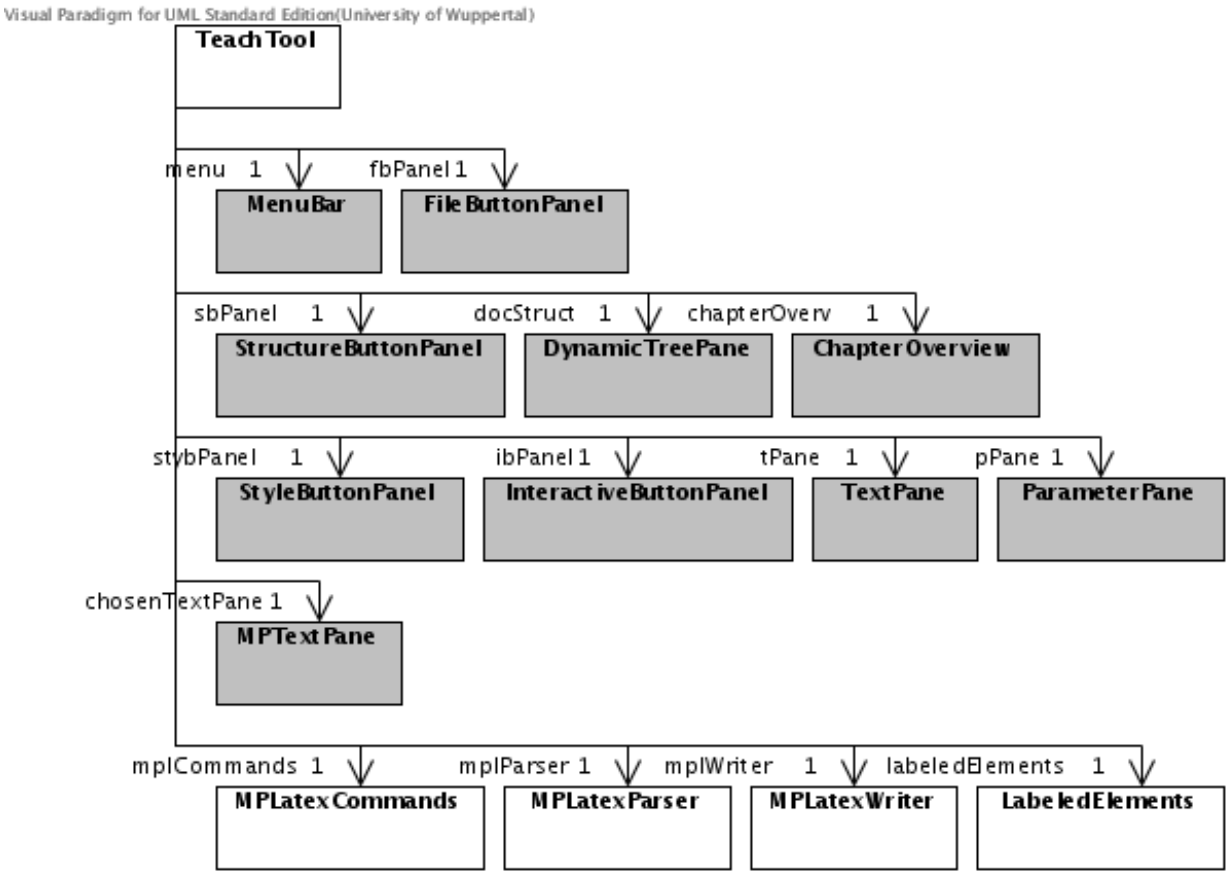


Abbildung 10.2: UML-Diagramm

10.1.2 Fließtext Komponenten

Die Darstellung von fettgedrucktem, farbigem und kursivem Text geschieht in *MPTextPane*, das von JTextPane erbt, wie von anderen Textverarbeitungsprogrammen gewohnt durch Modifizierung des Textstils.
Multiple Choice Aufgaben können im Fließtext jedoch nur durch das Einfügen spezieller Komponenten ansprechend dargestellt werden. Es gibt daher bei teachTool spezielle Klassen, welche die Darstellung im Text übernehmen. Sie implementieren das Interface *InTextComponent* und damit die folgenden Methoden:

- getLatexText()
  regelt, wie die Komponente durch LaTeX-Befehle repräsentiert wird
- setSelected(boolean selected, *MPTextPane* tPane)
  passt das Aussehen der Komponente an, wenn sie im Fließtext ausgewählt wird
- setCaretBehindPanel()
  setzt den Cursor im Text hinter die Komponente, wenn diese angeklickt wird
- openEditor()
  beim Anklicken der Komponente öffnet sich ein spezielles Editor-Fenster, in dem die Attribute der Komponente verändert werden können

In Tabelle 10.1 ist die Zuordnung zwischen Fließtext-Komponenten und Editor-Klassen zu sehen.

Name	*InTextComponent*	Editor
(z.B. Answer Field)	*MPButtonPanel*	(*ParameterPane*)
Puzzle	*PuzzlePanel*	*PuzzleEditor*
(Math.) Formula	*HotEqnPanel*	*HotEqnEditor*
Multiple Choice	*MultipleChoicePanel*	*MultipleChoiceEditor*
Hotspot	*HotspotPanel*	*HotspotEditor*
HTML	*OnlyInHtmlPanel*	*OnlyInHtmlEditor*
Applet	(*MPButtonPanel*)	*AppletPanel*
Table	*MPTable*	
Film		*FilmEditor*

Tabelle 10.1: Fließtext-Komponenten und Editor-Klassen

Jeder LaTeX-Befehl, der nicht durch Formatierung des Textes (z.B. textbf) oder durch eine spezielle Fließtext-Komponente dargestellt wird, erscheint im Text als *MPButtonPanel*. Das ist ein JPanel mit einem Button, der entweder ein Icon oder

den Namen des Befehls anzeigt. Besitzt der Befehl Parameter, werden diese gegebenenfalls in einem Label hinter dem Button angezeigt.
Als Editor fungiert bei *MPButtonPanel ParameterPane*, welches in einem modalen Fenster erscheint.
Tabellen benötigen keinen Editor, sie können über ein Kontextmenü angepasst werden.
Ein Film ist kein Fließtext-Objekt, sondern ein Abschnitt im Strukturbaum. Er kann über den in der Tabelle angegebenen Editor modifiziert werden.

10.1.3 LaTeX Umwandlung

Beim Öffnen der LaTeX-Datei einer Lerneinheit wird deren Struktur über *MPLatexParser* eingelesen. *MPLatexParser* sucht bei Einlesen nach LaTeX-Befehlen, die mit '\' beginnen. Wird ein Befehl gefunden, dann werden die Informationen aus *MPLatexCommands* (siehe Kapitel 3.2) verwendet, um die entsprechende Anzahl von Parametern einzulesen. Das Speichern der Informationen über die LaTeX-Befehle geschieht in Objekten der Klasse *MPElement* in folgenden Variablen:

```
String type;
String[] parameter = new String[20];
String[] didParameter = new String[20];
String text = "";
```

Die Zuordnung geschieht, abhängig davon ob es sich um eine LaTeX-Environment handelt oder nicht, auf folgende Art und Weise:

```
\begin{type}{parameter1}{parameter2}{...}
  text
\end{type}
```

bzw.

```
\type{parameter1}{parameter2}{...}
{
  text
}
```

Die didaktischen Attribute sind als Kommentare in den LaTeX-Dateien bei den entsprechenden Befehlen gespeichert.
Jedes *MPElement*-Objekt wird in einer DefaultMutableTreeNode in den Strukturbaum (*DynamicTree*) eingefügt. Dabei wird berücksichtigt, dass sich eine Seite in einem Kapitel und ein Textabschnitt in einer Seite befindet (siehe Abbildung 10.3).

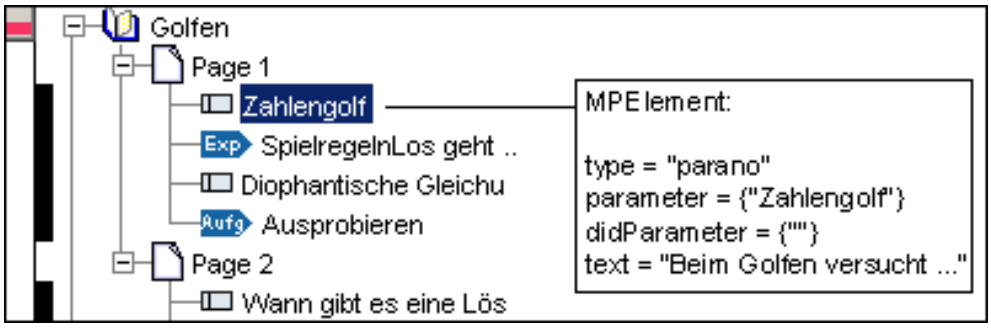


Abbildung 10.3: MPElement

Sobald ein Knoten im Strukturbaum angeklickt wird, werden die Parameter und der Text, die sich im *MPElement*-Objekt befinden, auf der teachTool Oberfläche in *MPTextPane*s und *MPTextField*s, die in *TextPane* bzw. *ParameterPane* eingebettet sind, angezeigt (siehe Abbildung 10.4).

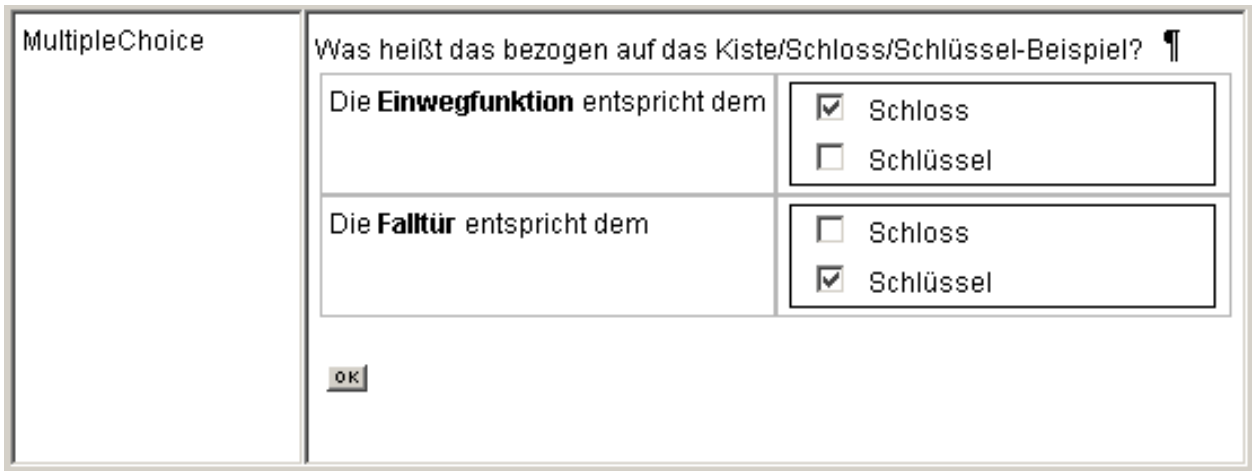


Abbildung 10.4: Zwei *MPTextPane*s mit Fließtext-Komponenten: *MPTable*, *MPButtonPanel* und *MultipleChoicePanel* im *TextPane*

Dazu müssen die Strings im *MPElement*-Objekt durch *MPLatexTranslator* umgewandelt werden. Es findet die in Abschnitt 10.1.2 beschriebene Textformatierung und das Einfügen von Fließtext-Komponenten statt.

Wenn in einem *MPTextPane* Veränderungen vorgenommen werden, muss der Eintrag im *MPElement*-Objekt aktualisiert werden. Die Rückumwandlung in LaTeX-Code übernimmt die Methode getTextElements() in *MPTextPane*. Sie wandelt die „CharacterElements“ von *MPTextPane* einen nach dem anderen um. Ist ein „CharacterElement“ z.B. eine Fließtext-Komponente, so gibt diese den zu verwendenden LaTeX-Code über die Methode getLatexText() zurück.

Das Speichern einer Lerneinheit übernimmt die Klasse *MPLatexWriter*. Diese durchläuft den Strukturbaum und schreibt die einzelnen Knoten in eine LaTeX-Datei.

10.1.4 HTML Konvertierung

Die Übersetzung der Lerneinheiten in HTML-Seiten hat Ähnlichkeit mit dem Speichern im LaTeX-Format.
MPLatexHTMLWriter durchläuft den Strukturbaum und legt entsprechende HTML-Seiten an. Jeder Abschnitt in einer Seite wird in HTML-Code übersetzt und in der momentan geöffneten HTML-Datei gespeichert. Beim Konvertieren von LaTeX nach HTML ist die Klasse *MPLatexHTMLCommands* behilflich. Diese ist vergleichbar zur Klasse *MPLatexCommands* (siehe Kapitel 3.2) aufgebaut, sie beschreibt für jeden LaTeX-Befehl, wie er in HTML präsentiert werden soll.
Das Erscheinungsbild der Internetseiten kann in teachTool über Extras → Design Options beeinflusst werden. Das Optionenfenster wird durch die Klasse *DesignOptions* gebildet.
Während der HTML Konvertierung wird in teachTool ein Fortschrittsbalken angezeigt. Dafür läuft die Übersetzung auf einem gesonderten Thread, der durch die Klassen *HTMLTask* und *SwingWorker* realisiert wird. Diese beiden Klassen sind zum größten Teil dem Sun Tutorial [Sun] Beispiel ProgressBarDemo entnommen.

Die Erzeugung des SCORM Exports erfolgt ebenfalls über *MPLatexHTMLWriter*. Es werden dabei hauptsächlich Unterschiede bei Interaktionen und das Erstellen der Datei imsmanifest.xml berücksichtigt.

10.1.5 Bearbeiten der Grundinformationen

Über den Button „General Information“ kann in teachTool ein Dialogfenster geöffnet werden, in dem grundlegende Eigenschaften einer Lerneinheit, wie z.B. das Bild für die Titelseite, der Name der Lerneinheit und die Namen der Kapitel, festgelegt werden können. Dieser Dialog besitzt drei Register, die durch die Klassen *GeneralChapterOverview*, *GeneralModuleInformation* und *GeneralPageInformation* bestimmt sind.

10.1.6 Wizard für Diashows

Diashows werden über einen Wizard in drei Schritten erstellt und bearbeitet. Der Button „edit Slideshow“ öffnet über *SlideshowActionListener* nacheinander die benötigten Dialogfenster. Deren Inhalt legen die Klassen *SlideshowEditorPanel1*, *SlideshowEditorPanel2* und *SlideshowEditorPanel3* fest.

10.1.7 Didaktische Benutzerführung

Durch Drücken des Buttons „Didactics Check“ wird eine Analyse der Lerneinheit nach didaktischen Gesichtspunkten über die Klasse *DidacticsCheck* durchgeführt (siehe Kapitel 4). Anpassungen der didaktischen Prüfung können durch *DidacticsCheckOptions* durchgeführt werden.
Eine weitere Übersicht über eine Lerneinheit nach didaktischen Kriterien liefert die „Didactics Documentation“, die in der Klasse *DidacticsDocu* erstellt wird.

10.1.8 FileChooser

ImagePreview und *MPFileFilter* sind Hilfsklassen für den JFileChooser, sie sind dem Sun Tutorial [Sun] Beispiel FileChooserDemo2 entnommen. *ImagePreview* erlaubt die Vorschau von Bildern im FileChooser. *MPFileFilter* ermöglicht die Auswahl von Dateien mit festgelegten Endungen.

10.1.9 Hilfsklassen

- *FileUtils*
  Methoden für das Kopieren Löschen und Umbenennen von Dateien und Verzeichnissen

- *IniFile*
  Laden und Speichern der INI-Datei, die Benutzereinstellungen für teachTool enthält

- *LabeledElements*
  Verwaltet Informationen zu allen *MPElement*s, die als Parameter ein Label besitzen

- *LatexFormulaeOptions*
  Optionenfenster zum Festlegen des LaTeX-Ordners, z.B. für die Erstellung von Formeln

- *MPButton*
  JButton mit einem speziellen Rand

- *SearchDialog*
  Dialogfenster für die Suche

- *SpellChecker*
  Rechtschreibprüfung

- *TablePanel*
  JPanel mit einer Tabelle und fünf Buttons zum Auf- und Abbewegen, Einfügen und Löschen einzelner bzw. aller Tabelleneinträge, wird z.B. in *PuzzleEditor* und *HotspotEditor* verwendet

10.2 Erweiterbarkeit von teachTool

In diesem Abschnitt wird beschrieben, wie in teachTool weitere Grundelemente (z.B. ein Kreuzworträtsel) ergänzt werden können.
Jedes neue Grundelement muss in den Klassen *MPLatexCommands* und *MPLatexHTMLWriter* hinzugefügt werden.
In *MPLatexCommands* sind Informationen, wie die Parameter und das zu verwendende Icon gespeichert (siehe Kapitel 3.2).
In *MPLatexHTMLCommands* wird die Präsentation eines Grundelements in HTML festgelegt.
Bei beiden Klassen muss die Konstante NUMOFCOMMANDS angepasst werden.
Damit ist im einfachsten Fall schon alles erledigt, da das Grundelement im Fließtext durch *MPButtonPanel* dargestellt wird.

Soll jedoch eine spezielle Fließtext-Komponente verwendet werden, muss eine Klasse, die z.B. von JPanel erbt und das Interface *InTextComponent* implementiert, angelegt werden. Für die Methode openEditor(), die in *InTextComponent* vorgeschrieben wird, ist die Implementierung eines Dialogs mit z.B. einem JPanel als Editor notwendig. Die Umwandlung in LaTeX-Code geschieht über die Methode getLatexText(), die ebenfalls in *InTextComponent* abstrakt angelegt ist. Damit die Fließtext-Komponente in einem *MPTextPane* angezeigt werden kann, muss sie in *MPLatexTranslator.latexToText(...)* eingetragen werden. Ähnlich ist auch die Symbolleiste, auf der das Grundelement erscheinen soll, zu modifizieren. Hier muss über actionPerfomed(ActionEvent) vermerkt werden, dass die Fließtext-Komponente beim Drücken des entsprechenden Buttons in ein *MPTextPane* eingefügt wird. In *DidacticsCheck* und *SpellChecker* werden die Wörter des Grundelements gezählt bzw. auf korrekte Rechtschreibung überprüft. Es muss festgelegt werden, welche Wörter berücksichtigt werden sollen. Im Fall eines Kreuzworträtsels soll dieses in *DidacticsCheck* als Applet und damit als Interaktionsform berücksichtigt werden. Ferner muss gegebenenfalls ein Ordner, der die Java-Klassen für das Kreuzworträtsel-Applet beinhaltet, bei der Konvertierung nach HTML in den Modulordner kopiert werden. Das sollte in *HTMLTask* eingetragen werden.

Auch wenn keine spezielle Fließtext-Komponente benötigt wird, kann es sein, dass in *ParameterPane*, welches von *MPButtonPanel* als Editor benutzt wird, Erweiterungen implementiert werden müssen.
Wenn z.B. ein Parameter eines Grundelements vorsieht, dass eine Datei über einen

FileChooser ausgewählt wird, empfiehlt es sich, diesen Parameter „Filename“ zu nennen. In *ParameterPane* wird automatisch ein Button zum Öffnen eines FileChoosers hinzugefügt. Alle für eine Lerneinheit relevanten Dateien werden von teachTool in den Modulordner kopiert, falls sich diese nicht schon darin befinden. Daher muss gegebenenfalls in *ParameterPane.openFileChooser(int)* und *TeachTool.chooseFile(...)* festgelegt werden, in welchen Unterordner die Dateien kopiert werden sollen.

10.3 Funktionsweise der Konvertierung nach PDF

Die Konvertierung einer Lerneinheit nach PDF geschieht mit Hilfe von pdflatex und wird von der Methode *MenuBar.pdfOutput(...)* durchgeführt. Zuvor werden die benötigten Dateien mittels *MenuBar.copyFilesToTempDir(...)* in ein temporäres Verzeichnis kopiert, wobei z.B. GIF-Dateien in PNG-Dateien umgewandelt werden[2]. In *MenuBar.pdfOutput(...)* wird die TEX-Datei der Lerneinheit durch den *MPLatex-Writer* im temporären Verzeichnis gespeichert, wobei spezielle Änderungen für die Verarbeitung mit pdflatex vorgenommen werden. Dabei werden z.B. alle Tabellen so modifiziert, dass die Spalten dieselbe Breite besitzen, um Umbrüche in Tabellenzellen zu ermöglichen. Anschließend wird pdflatex ausgeführt, die erzeugte PDF-Datei in den Modulordner kopiert, das temporäre Verzeichnis gelöscht und die PDF-Datei in einem Betrachter-Programm geöffnet.

[2] pdflatex kann keine GIF-Dateien verarbeiten.

Kapitel 11

Einsatz und Evaluation

(Lehramts-) Studierende setzen teachTool zum Erstellen von Modulen für ihre Examensarbeit bzw. in Seminaren ein.
Ferner arbeiten auch Schülerinnen und Schüler der Stufen 10 - 11 im Rahmen von Praktika mit teachTool.
Des Weiteren wird teachTool von den Autoren der MathePrisma- und der OptiV-Entwicklergruppe (Universitätsmitarbeiter und Professoren) verwendet.
In Zukunft sollen auch Lehrer zum Autorenkreis hinzukommen. Sie können fertige MathePrisma-Module an ihre schulischen Bedürfnisse anpassen und eigene Lerneinheiten entwickeln.

Aufgrund des sehr großen Nutzerkreises wurde bei der Entwicklung von teachTool besonders auf Kriterien der Usability Wert gelegt. Es stellen sich dabei Fragen wie:

- Wie viel Eingewöhnungszeit benötigt ein teachTool-Autor?
- Ist die Programmoberfläche intuitiv bedienbar?
- Ist der Aufbau der GUI leicht zu überblicken?
- Welche Kenntnisse in HTML und JavaScript werden vorausgesetzt?

Ferner geht es auch um den Funktionsumfang:

- Besitzt teachTool alle Funktionalität, die der Autor erwartet?
- Ist das Design der Lerneinheiten hinreichend anpassbar?
- Wie nehmen Autoren die Unterstützung des Programms bei der Einhaltung didaktischer Regeln war?

Auf der technischen Seite interessiert:

- Welche technischen Probleme (z.B. Programmabsturz) gibt es?
- Kann teachTool tatsächlich problemlos auf verschiedenen Betriebssystemen verwendet werden?
- Sind die mit teachTool erstellten Lerneinheiten wirklich auf allen (gängigen) Browsern einsetzbar?

Die Evaluation von teachTool erfolgte formativ. Das Programm wurde also fortlaufend überarbeitet mit Berücksichtigung der zum jeweiligen Zeitpunkt bekannten Kritikpunkte. Somit arbeiteten die drei im folgenden genauer beschriebenen Autorengruppen mit unterschiedlichen und sich kontinuierlich verändernden / verbessernden Versionen der Software.

11.1 Universität Wuppertal

Im Wintersemester 2004/2005 wurde eine erste Version von teachTool in dem Seminar „MathePrisma“ im Fachbereich Mathematik und Naturwissenschaften an der Bergischen Universität Wuppertal eingesetzt.
An der Veranstaltung nahmen 7 Studierende des Lehramts für Sekundarstufe I bzw. I + II teil, die sich im 5. - 7. Semester befanden.[1]
Lernziel der zwei Semesterwochenstunden umfassenden Veranstaltung war der Erwerb von Medien- und Vermittlungskompetenz. Zu diesem Zweck erstellten die Studierenden MathePrisma-Module zu selbstgewählten Themen der Mathematik.
Bestandteil der Veranstaltung war eine Einführung in das Design- und didaktische Konzept von MathePrisma, in Bildbearbeitung und die Verwendung von dynamischer Geometriesoftware (am Beispiel Cinderella [Kor]).

Am Anfang des Seminars wurden Fragebögen ausgegeben, um das Vorwissen der Studierenden einordnen zu können.
Das Ergebnis der Befragung war, dass die Studierenden sehr viele Kenntnisse im Umgang mit Standardsoftware (wie z.B. Word), mit dem Internet und beim Bearbeiten von Bildern besaßen. Im Bereich des Erstellens von Internetseiten, bei Programmiersprachen und im Verwenden von LaTeX-Notation war jedoch kaum Vorwissen vorhanden.

Der zweite Fragebogen wurde verteilt, nachdem die Studierenden kurze Zeit mit teachTool gearbeitet hatten. Fragen bei denen es darauf ankam, Bewertungen abzugeben, wurden durch Ankreuzen der „Noten“ ++, +, +−, − bzw. −− beantwortet

Kriterium	++	+	+−	−	−−
Benutzerfreundlichkeit: teachTool		6	2		
Übersichtlichkeit: teachTool	3	2	2	1	
Benutzerfreundlichkeit: Slideshow-Editor	2	2	3	1	
Benutzerfreundlichkeit: Math-Editor		3	3	2	
Benutzerfreundlichkeit: MultipleChoice-Ed.	1	5	2		

Tabelle 11.1: Evaluation im Seminar „MathePrisma“, Fragebogen 2

(siehe Tabelle 11.1). Die Bewertungen untermauern den Eindruck aus der Veranstaltung, dass die Studierenden sehr schnell lernten, mit teachTool zu arbeiten. Selbst die Meinungen zum Formel-Editor (Math-Editor) (siehe Kapitel 7) sind sehr positiv, wenn man bedenkt, dass die Studierenden sich nicht mit LaTeX auskannten.
Als wichtige multimediale Elemente wurden in erster Linie Diashow, Texteingabe-Aufgaben und Multiple Choice genannt.
Im frei ausfüllbaren Bereich, in dem nach fehlenden multimedialen Elementen gefragt wurde, nannten die Studierenden zum Teil Flash- und Sounddateien. Gründe, die gegen die Verwendung von Flash sprechen, sind die Tatsache, dass im Browser ein Plug-In benötigt wird und dass die Dateien aufgrund ihrer Größe recht lange Übertragungszeiten besitzen. Die wachsende Zahl an Breitbandverbindungen und die immer häufigere Verwendung von Flash im Internet zeigen hingegen, dass es sinnvoll wäre, teachTool in Zukunft in diesem Bereich zu erweitern. Gegen den Einsatz von Sounds spricht der parallele Einsatz der Lerneinheiten auf verschiedenen Computern in einem (Klassen-)Raum. Wenn über Lautsprecher verschiedene Sounddateien abgespielt werden, führt das zu einer Störung der Lernatmosphäre. Auf der anderen Seite sollte gerade in den Sprachwissenschaften auf das Zurverfügungstellen von Hörproben nicht verzichtet werden. Es bietet sich deshalb an, den Funktionsumfang von teachTool in den folgenden Versionen in diesem Bereich zu erweitern.
Momentan können Flash- und Sounddateien in teachTool eingebaut werden durch das direkte Hinzufügen von HTML-Codes wie z.B.

```
<embed src="datei.swf" quality="high" scale="exactfit"
  menu="false" bgcolor="#000080" width="600" height="400"
  swLiveConnect="false" type="application/x-shockwave-flash"
  pluginspage="http://www.macromedia.com/shockwave/download/
               download.cgi?P1_Prod_Version=ShockwaveFlash">
```

Weitere Kritikpunkte, wie Probleme bei der Bearbeitung von Diashows oder das Fehlen eines Papierkorbs zum Wiederherstellen gelöschter Knoten im Strukturbaum

[1]Zwei der Studierenden erstellten später ein MathePrisma-Modul im Rahmen ihrer Staatsexamensarbeit.

haben zur Weiterentwicklung von teachTool beigetragen.
Hilfefunktionen wünschten die Studierenden für den Slideshow-Editor, den Formel-Editor (Math-Editor) und das Einfügen von Strukturelementen. Eine Online-Hilfe wurde daraufhin für teachTool implementiert.
Als technisches Problem stellte sich die Darstellung von Formeln in den HTML-Seiten der Lerneinheiten durch HotEqn-Applets heraus. Es gab dabei häufig Browserabstürze. Dieses Problem wurde durch das Einbinden von Grafikdateien anstelle von HotEqn behoben. Der Wunsch der Studierenden nach einer deutschsprachigen Version wurde in der aktuellen Version realisiert.

Den dritten Fragebogen, der am Ende des Seminars herausgegeben wurde, beantworteten lediglich drei Studierende. Sie bestätigten den positiven Gesamteindruck von teachTool bezüglich Benutzerfreundlichkeit und Übersichtlichkeit.
Auf die Frage „Wie sehr war Ihnen Ihr mathematisches Vorwissen bei der Bedienung von teachTool behilflich?“ wurde zweimal mit − und einmal mit +− geantwortet. Es ist daher interessant zu untersuchen, ob sich die in dieser Evaluation gemachten Beobachtungen auch auf Autoren anderer Fachgebiete übertragen lassen.
Der „Didactics Check“ wurde nur von 2 der 3 Studierenden verwendet. Hier lassen sich daher kaum repräsentative Aussagen treffen.
Sehr positiv waren die Antworten auf die Fragen, ob sich die Befragten vorstellen können, als Lehrer mit teachTool Module zu erstellen bzw. auf ihre Bedürfnisse im Unterricht anzupassen. Bis auf eine Antwort, bei der kein Kreuzchen gesetzt wurde, antworteten die drei Befragten bei beiden Fragen mit ++.

11.2 OptiV

Seit Mitte des Jahres 2005 wird teachTool von den Mitarbeitern im Projekt OptiV - Erschließung von Entscheidungs- und Optimierungsmethoden für die Anwendung im Verkehr - eingesetzt [BFB].

OptiV ist ein vom Bundesministerium für Bildung und Forschung (bmb+f) gefördertes Projekt folgender Lehrstühle des Verkehrswesens als Hauptauftragnehmer:

- Fachgebiet Verkehrsplanung und Verkehrstechnik (FGVV), Technische Universität Darmstadt, Univ.-Prof. Dr.-Ing. Manfred Boltze
- Institut für Verkehrswirtschaft, Straßenwesen und Städtebau (ivh), Universität Hannover, Univ.-Prof. Dr.-Ing. Bernhard Friedrich
- Lehrstuhl für Wirtschaftsinformatik und Operations Research (WINFOR), Rheinisch-Westfälische Technische Hochschule (RWTH) Aachen, Univ.-Prof. Dr. rer. pol. habil. Michael Bastian

Ziel des Projekts ist die Verbreitung und Intensivierung der universitären Ausbildung von Nachwuchs-Ingenieuren im Bereich der Entscheidungs- und Optimierungsmethoden. In dem Projekt sollen der allgemeine Entwicklungsstand von Entscheidungs- und Optimierungsmethoden dargestellt, die Anwendung von Entscheidungs- und Optimierungsmethoden im Verkehrswesen für Lehre und Praxis dokumentiert, die Anwendungsmöglichkeiten neuer Entscheidungs- und Optimierungsmethoden im Verkehrswesen und der weitere Entwicklungsbedarf erarbeitet und die so gewonnenen Ergebnisse verbreitet werden.
Hierzu sollen im Rahmen von OptiV die Möglichkeiten einer elektronischen Aufbereitung und Verbreitung der Ergebnisse genutzt werden. Die Ergebnisse aus OptiV sollen in der universitären Lehre wie in der außeruniversitären Fortbildung zum Einsatz kommen. Fragen nach der didaktischen Qualität sind deshalb von herausragender Bedeutung.
Die Projektlaufzeit begann am 01.05.2004 und endet am 31.07.2006.

Auf der Suche nach Projekten, die als Beispiel für die in OptiV vorgesehene Erzeugung von E-Learning-Einheiten dienen konnten, hat das OptiV-Konsortium Kontakt zu den MathePrisma-Verantwortlichen hergestellt.
Bei der Auswahl eines für OptiV geeigneten Autorensystems wurden in einer von M. Bohlinger durchgeführten Analyse bei verschiedenen Autorensystemen folgende Kriterien überprüft:

- Einarbeitungsaufwand (einfache Bedienung)
- Eingabeaufwand (Eingaberoutinen, Dauer zur Erstellung einer Seite)
- Kosten Programm
- Betreuung durch Anbieter
- Kosten Betreuung
- Erweiterbarkeit (für zusätzliche Fallbeispiele)
- Plattformunabhängigkeit
- Änderungsmöglichkeiten
- Einheitliche Gestaltung
- Darstellungsmöglichkeiten
- Übertragbarkeit (Internet, CD, Druck)
- Flexible Nutzbarkeit
- Entwicklungsstand
- Erfahrung des Anbieters bei der Erstellung ähnlicher Inhalte

Nach einem Vergleich der von den Anbietern der Systeme gemachten Angaben und da die Grundprinzipien von MathePrisma in den Bereichen Didaktik und Design sehr überzeugten, entschloss man sich im Projekt OptiV, teachTool als Autorenwerkzeug zu verwenden.
Durch eine zusätzliche Finanzierung von Seiten des bmb+f konnte an der Universität Wuppertal eine halbe Mitarbeiterstelle für 8 Monate finanziert werden. Im Rahmen dieser Stelle wird die Funktionalität von teachTool erweitert, um die Anforderungen, die sich in OptiV ergeben, optimal erfüllen zu können. Insbesondere betrifft dies z.B. die Verbesserung des Exports der Lerneinheiten nach PDF oder die Erweiterung der Design-Optionen für die Generierung der Internetseiten.

Im Projekt OptiV wurden Fragebögen zu teachTool (siehe Seite 108) von 12 Beteiligten beantwortet.

Der erste Teil der Fragebögen beschäftigt sich mit dem Vorwissen der Befragten. Sie gaben fast ohne Ausnahme an, umfangreiche Erfahrung im Verwenden von Textverarbeitungsprogrammen, im Umgang mit dem Internet und bei Programmiersprachen zu besitzen. Kenntnisse über HTML und LaTeX besitzt nur ca. die Hälfte der Befragten.

Besonders bedeutend im Hinblick auf die Verwendbarkeit (Usability) von teachTool sind die beiden ersten Fragen im Bereich „Technik“. 8-mal wurde die Eingewöhnungszeit für das Programm als niedrig im Vergleich zu Standardanwendungen wie Word, Excel, ... eingeschätzt. Auch die Benutzerfreundlichkeit von teachTool wurde mit 4 ++, 5 + und 3 +− genauso wie die Übersichtlichkeit (4 ++ und 8 +) sehr positiv bewertet.
Die Benutzerfreundlichkeit beim Erstellen von Multiple Choice Aufgaben und Diashows wurde als gut eingestuft. In diesem Punkt wird sehr deutlich, dass teachTool seit dem Einsatz und der Evaluation im Seminar an der Universität Wuppertal in einigen Bereichen verbessert wurde. So traten bei der im Projekt OptiV verwendeten Version des Systems fast keine Probleme beim Erstellen von Diashows auf. Leichte Probleme gab es bei der Eingabe mathematischer Formeln (2 ++, 3 +, 6 +−, 1 −), was auch durch das Anfragen einer erweiterten Online-Hilfe für den Formel-Editor zum Ausdruck kommt. Angesichts der Tatsache, dass die Hälfte der Befragten keine Vorkenntnisse bei LaTeX angaben, ist die Bewertung des Formel-Editors aus Sicht des Autors recht zufriedenstellend. Es gibt hier jedoch sichtlich ein Potential die Qualität von teachTool weiter zu verbessern.
Der Großteil der Antworten zeigt, dass eine klare Trennung von Struktur und Inhalt durch die Programmoberfläche vermittelt wird und dass die Lerneinheiten im Browser dem entsprechen, was beim Bearbeiten mit teachTool vermittelt wird.
Bei den interaktiven Elementen werden auf den beantworteten Fragebögen Diashows und Applets als besonders wichtig herausgestellt. Gerade bei diesen Elementen besitzt teachTool umfangreiche Assistenten (Wizards), die das Erzeugen bzw. Einfügen

unterstützen. Ferner werden Multiple Choice Aufgaben und Filme als von großer Bedeutung eingeschätzt.
Auf die Frage, ob es wichtig wäre, weitere Strukturierungselemente in teachTool hinzuzufügen, gab es bei den Antworten einige Befürwortungen. Es wurde jedoch nur einmal konkret genannt, dass ein Abschnitt für Beispiele fehle.
Die Verwendung von Computer-Algebra-Systemen war für die Befragten von geringer Bedeutung.
Des Weiteren wurde eine Erweiterung der Hilfefunktion, besonders zum Formel-Editor und dem Einfügen von Strukturelementen und die Verbesserung der Druckversion und der Bearbeitung von Tabellen vorgeschlagen.

Die didaktische Benutzerführung wurde spezielle durch Fragen nach dem „Didactics Check“ überprüft. Die Bedeutung des „Didactics Checks“ wurde sehr unterschiedlich eingestuft. Allerdings gaben auch nur 4 der Befragten an, diese Funktion verwendet zu haben. Von diesen Vieren wurde die Bedeutung dreimal mit ++ und einmal mit − bewertet. Die folgenden drei Fragen, die Wissen über die vom „Didactics Check“ gegebenen Hinweise voraussetzt, beantworteten fünf Personen. Die Einschätzung, auf welche Art und Weise der Check bei der Stoffsequenzierung unterstützen konnte, fiel sehr unterschiedlich aus (1 ++, 3 +, 1 −, 1 −−). Die Frage „Wie bewerten Sie die zugrundeliegenden Kriterien?“ wurde recht positiv mit 3 + und 2 +− beantwortet. Aufgrund der Hinweise des „Didactics Check“ gaben alle fünf Personen an, Seiten umstrukturiert und Informationen auf den Nebenpfad verschoben zu haben. Das deckt sich mit der Bemerkung eines OptiV-Mitarbeiters zu Beginn der Erstellung der Internetmodule: „Die bisher erstellten Seiten sind aber sehr textlastig. Das zeigt, wie wichtig die didaktische Aufbereitung der Themen ist.“ Ferner gaben jeweils vier der Befragten an, nach „Didactics Check“-Ratschlägen Texte gekürzt und Bilder hinzugefügt zu haben.

Die Möglichkeit zur Erzeugung einer Printversion von Lerneinheiten mit teachTool wurde durch die Kooperation im Projekt OptiV sehr vorangetrieben. Bis auf wenige technische Probleme bescheinigen die OptiV-Mitarbeiter dem Aussehen der mit teachTool generierten PDF-Dateien eine gute Qualität.

11.3 Universität Mainz

An der Johannes Gutenberg-Universität Mainz wurde im Fachbereich Mathematik im Sommersemester 2005 teachTool in dem Seminar „Modellierung und Visualisierung“ von Studierenden verwendet.
Thema des Seminars war die mathematische Modellierung realer Probleme, deren (numerische) Lösung und die Visualisierung der Ergebnisse im Rahmen eines dazu zu erstellenden Internetauftritts in einer Form, die der von MathePrisma-Modulen angelehnt ist. Wie auch im Projekt OptiV waren die Seminarleiter von den Prinzipien

von MathePrisma überzeugt. Daher wurde zur Vereinfachung des Entwicklungsaufwands von den Studierenden teachTool eingesetzt.

An dem Seminar nahmen 25 Studierende teil, die jedoch nicht alle mit teachTool arbeiteten. Die Befragung der Studierenden konnte erst ca. ein Dreivierteljahr nach Beendigung der Veranstaltung durchgeführt werden. Von den 8 Personen, die die Fragebögen (siehe Seite 108) beantworteten, gaben viele an, dass sie sich nicht mehr in allen Einzelheiten an die Benutzung des Programms erinnerten.

Fast alle Studieren erklärten, sich gut mit Textverarbeitung und dem Internet auszukennen. Nur wenige besaßen hingegen Kenntnisse im Erstellen von HTML-Seiten, in LaTeX und beim Programmieren.

Fünf der Befragten schätzten die Eingewöhnungszeit für teachTool als niedrig ein und sieben bewerteten die Benutzerfreundlichkeit mit +. Auch die Übersichtlichkeit des Programms wurde als gut bis befriedigend benannt.
Beim Erstellen von Diashows und mathematischen Formeln traten ein paar technische Probleme auf. So konnte z.B. in der verwendeten teachTool Version kein Einfluss auf die benutzen LaTeX-Pakete genommen werden, was bei der Eingabe von Formeln Schwierigkeiten hervorrief. Diese Probleme wurden jedoch bereits behoben. Die Gestaltung von Multiple Choice Aufgaben stellte sich als zufriedenstellend heraus.
Die Frage „Wie klar kommt die Trennung zwischen Struktur und Inhalt auf der Oberfläche von teachTool heraus?“ wurde mit 1 ++, 2 + und 3 +− beantwortet.
Die meisten Befragten meinten, dass sie beim Bearbeiten der Lerneinheiten gut erkennen konnten, wie die Module im Browser erscheinen.
Das interaktive Element, das für fast alle Studierende von Wichtigkeit war, sind Applets, gefolgt von Texteingabe-Antworten und Filmen.
Wie bei der Befragung der OptiV-Mitarbeiter gab es bei der Frage, ob es wichtig wäre, weitere Strukturierungselemente in teachTool hinzuzufügen, bei den Studierenden ein paar Befürworter. Auch hier wurde jedoch nur einmal konkret genannt, dass ein Abschnitt für Beispiele fehle.
Die Studierenden bemaßen der Verwendung von Computer-Algebra-Systemen nur geringe Bedeutung bei.
Aufgrund von technischen Problemen bei der Verwendung von Diashow- und Formel-Editor überrascht der Wunsch nach einer Hilfefunktion für diese beiden Elemente nicht. Diese Hilfefunktion wurde daraufhin erstellt.

Nur einer der Befragten gab an, den „Didactics Check“ benutzt zu haben. Daher gab es keine repräsentativen Ergebnisse zu dieser Funktion.

Die generelle Einschätzung der Seminarleiterin zu teachTool war wie folgt: „... die meisten Gruppen haben mit dem TeachTool gearbeitet und waren damit auch zufrieden. Ab und an gab es mal kleinere Probleme (z.B. mit der Diashow), aber insgesamt ist das Tool sehr selbsterklärend.“

Kapitel 12

Zusammenfassung

Dieses Kapitel gliedert sich – vergleichbar zum Beginn der Arbeit – in die Bereiche technische Grundelemente, Struktur, Didaktik und Design. Im letzten Abschnitt werden weiter Funktionselemente von teachTool abschließend betrachtet.

12.1 Technische Grundelemente

teachTool hat sich als ein Autorensystem erwiesen, das im Funktionsumfang mit den marktführenden Systemen vergleichbar ist.
Insbesondere die Diashow, die sich in den Projekten MathePrisma und OptiV als äußerst vielseitige und beim E-Learning hilfreiche Interaktionsform herausgestellt hat, kann mit teachTool sehr komfortabel erstellt werden. Auch die Antworten zu den Fragebögen der Evaluation unterstreichen die große Bedeutung von Diashows. Deren Erstellung wird jedoch von den untersuchten Programmen ToolBook, Lectora und LERSUS nicht unterstützt. Weitere Typen von Interaktionen, die nicht zum Standard vieler Autorensysteme gehören, sind MouseOver, bei dem beim Bewegen der Maus über ein Bild ein anderes Bild angezeigt wird und Applets, die mit Punkten versehen und in das Bewertungssystem integriert werden können.
Ein wichtiges Grundelement zur Erstellung von E-Learning-Modulen im Bereich der Mathematik und der Naturwissenschaften ist der eingebaute Formel-Editor. Er beruht auf LaTeX und kann daher sowohl mathematische Formeln verarbeiten als auch z.B. Strukturformeln aus der Chemie.
Die Gestaltungsfreiheit für individuelle Interaktionen ist bei teachTool sehr groß, da beliebiger HTML-Code in die Lerneinheiten eingefügt werden kann und Interaktionen hierüber mit JavaScript erschaffen werden können.
Einen Beitrag zur Verbesserung der didaktischen Qualität bieten in teachTool die Abschnittstypen für einleitenden/motivierenden Text, Definitionen/(Merk)-Sätze und Interaktionen. Sie fordern den Autor dazu auf, sich über die Sequenzierung der Module Gedanken zu machen und erleichtern dem Lernenden das Überblicken der

Inhalte und ihrer Bedeutung. Gleichzeitig erleichtert die Einschränkung des Funktionsumfangs z.B. bei den Textformaten dem Autor die Wahl, beim Finden geeigneter Darstellungsformen ohne die Lerneinheit zu überfrachten.
Ausbaufähig ist teachTool im Bereich multimedialer Elemente wie Flash, Video und Ton. Gründe, die gegen die Verwendung dieser Multimediaelemente sprechen, sind die Tatsache, dass im Browser ein Plug-In oder eine Verknüpfung zu einem externen Programm benötigt wird und dass die Dateien aufgrund ihrer Größe recht lange Übertragungszeiten besitzen. Die vielseitige Einsetzbarkeit dieser Elemente und die Tatsache, dass sie im Internet mit wachsender Zahl der Breitbandverbindungen immer mehr Anwendung finden, werden in den nächsten Versionen zu einer Weiterentwicklung von teachTool führen. Angeregt durch die Zusammenarbeit im Projekt OptiV kam ein Glossar aktuell hinzu. Die Erweiterung des Umfangs von Aufgabenformaten z.B. vom Typ Drag&Drop oder Kreuzworträtsel, die besonders beim Einsatz in den Sprachwissenschaften von Interesse sind, ist geplant.

12.2 Struktur

teachTool unterstützt die Anlage einer sequenziellen Struktur der Module. Alle hierfür notwendigen Navigationsmechanismen werden für die Internetseiten automatisch generiert. Auch das erleichtert den Autoren die Arbeit, da sie sich so auf das Einfügen der Inhalte konzentrieren können. Zusätzlich bietet teachTool Nebenpfadseiten an, die den linearen Lernpfad durch weiterführende Informationen bereichern und variabler gestalten. Diese Nebenpfadseiten führen den Lernenden jedoch stets wieder auf den Hauptlernpfad zurück. Es wird somit das Problem des „lost in hyperspace“ vermieden.
Eine weitere strukturelle Komponente bildet das Arbeitsblatt, welches für jede Lerneinheit als Zusatzmaterial z.B. für den Einsatz im Unterricht zur Lernzielkontrolle vorgesehen ist.
Die gesamte Struktur der Lerneinheiten wird in teachTool in einem Strukturbaum dargestellt. In diesem können durch Drag&Drop einfach Veränderungen am Aufbau vorgenommen werden.
Die in erster Linie linear angelegte Struktur steht hier anscheinend sehr im Gegensatz zu den Forderungen der konstruktivistischen Lerntheorie. Es sei aber darauf aufmerksam gemacht, dass Umfragen im Rahmen von MathePrisma zeigten, dass der Aufbau von Benutzern als hilfreich bei der Orientierung und nicht als einschränkend eingeschätzt wird. Außerdem kann über die Navigationsleiste und durch das Angebot weiterer Verweise darüber hinaus die Reihenfolge der besuchten Seiten individuell gewählt werden (vergleiche [Kri03], Seite 41).

12.3 Didaktik

Eine große Stärke von teachTool liegt in der didaktischen Benutzerführung. Durch den „Didactics Check“ kann der Autor jederzeit die Struktur seiner Lerneinheit z.B. nach handlungsorientierten Grundsätzen überprüfen. Es wird ferner ermittelt, ob Strukturelemente auch tatsächlich durch ihren Inhalt die ihnen zugeteilte Bestimmung erfüllen. Zum Beispiel muss ein interaktiver Abschnitt auch Interaktionen beinhalten. Ferner gibt der „DidacticsCheck“ dem Autor einen Überblick, ob der Umfang eines Moduls geeignet gewählt wurde oder ob genügend Interaktionen und Visualisierungen Verwendung finden. Die im „Didactics Check“ überprüften Kriterien können in einem speziellen Optionenfenster individuell angepasst werden.
Jeder Seiten-Abschnitt besitzt die didaktischen Metadaten Lernphase und Lernzeit. Diese unterstützen den Autor beim Umstrukturieren und dem Berücksichtigen des Gesamtaufwands. Sie sind zusätzlich interessant für Lehrende, die Module nach ihren Bedürfnissen modifizieren möchten. Aufgrund der Lernphaseneinteilung kann mittels der „Didactics Documentation“ eine Kurzübersicht der Lerneinheit zusammengestellt werden.
Ausbaufähig aus didaktischer Sicht ist die Punktevergabe und das Feedback bei der Auswertung von Aufgaben.

12.4 Design

Die Erfahrungen in der Zusammenarbeit mit dem Projekt OptiV haben gezeigt, dass die Anpassbarkeit des Designs der erzeugten Internetseiten einigen Spielraum zulässt. Es ist jedoch zu Überprüfen, ob eine größere Variabilität z.B. der Startseite, der Seitenköpfe, der Navigationsleiste oder der für die Seiten-Abschnitte gewählten Bilder in anderen Projekten von Vorteil sein könnte.

12.5 Weiterer Funktionsumfang

Der Funktionsumfang von teachTool umfasst das Erzeugen einer PDF-Version, die das Ausdrucken der Module, z.B. für die Fehlerkorrektur beim Erstellen oder das Nachbereiten der Inhalte ohne Computer, ermöglicht. Durch den SCORM-Export wird die zukünftige Verwendbarkeit und der Einsatz in Lernplattformen sichergestellt. Gerade im Bereich der Lehramtsausbildung kann teachTool als Lernprogramm eingesetzt werden, da es Überlegungen fördert und fordert, die für angehende Lehrende ebenso bei der Planung von Unterrichtsstunden von Bedeutung sind. Bei der Implementierung des Autorensystems in Java und der HTML-Konvertierung der Lernmodule wurde darauf geachtet, dass sowohl teachTool als auch die damit erstellten Internetseiten auf allen gängigen Betriebssystemen und Browsern zum Einsatz

kommen können. Ein Problem bei der Verwendung von Autorensystemen insbesondere in Bildungseinrichtungen wie Schulen und Universitäten sind die zum Teil sehr hohen Kosten dieser Programme. teachTool wird daher in Zukunft kostenlos zum Herunterladen angeboten. Als letzten Pluspunkt von teachTool soll die vielerorts bewährte leichte Bedienbarkeit und Überschaubarkeit des Programms erwähnt werden, die den Einsatz für verschiedene Autorenkreise wie professionelle E-Learning-Entwickler, Hochschulangestellte, Studierende, Lehrer und Schüler ermöglicht.
Eine deutschsprachige Version des Programms wurde bereits umgesetzt und trägt dazu bei die Bedienbarkeit noch weiter zu verbessern. Die Übersetzung ins Französische ist geplant.

Danksagung

Mein Dank gebührt Herrn Prof. Dr. A. Frommer für sein intensives Interesse am Projekt MathePrisma und seine Unterstützung bei der Erstellung von teachTool. Vielen Dank an Prof. Dr. M. Stein für seine Hilfe und die Übernahme des Korreferates. Besonderer Dank für ihre Motivation, diese Arbeit zu schreiben, gilt meinen tollen Kollegen, insbesondere Stefanie Krivsky-Velten und Thomas Beelitz. Einen wichtigen Anteil an der Verbesserung von teachTool hat die sehr gute Zusammenarbeit mit dem Projekt OptiV. Danke für jede stets freundliche Kritik. Stellvertretend für viele Studierende möchte ich Gerd Heischkamp danken. Für das schöne Logo bedanke ich mich bei Nicole Bölt. Herzlichen Dank an meine Frau Verena und meine Familie, die mir geholfen haben, wenn die Motivation in den Keller ging. Danke dabei auch an Luis für seine regelmäßige Aufmunterung.

Anhang A

Fragebögen

In diesem Kapitel befinden sich vier Fragebögen (siehe Kapitel 11 „Einsatz und Evaluation“). Die ersten drei wurden nacheinander im MathePrisma-Seminar verteilt. In jedem Ankreuzfeld ist die Anzahl der jeweils gesetzten Kreuzchen eingetragen. Der vierte beginnt auf Seite 108 und wurde für die Befragung der Beteiligten im Projekt OptiV und der Teilnehmer im Seminar an der Universität in Mainz verwendet. Die Zahl vor dem Semikolon in den Ankreuzfeldern bezieht sich auf die Umfrage im Projekt OptiV die dahinter auf die im Seminar an der Universität in Mainz.

(Name oder Geburtsdatum der Mutter)

MathePrisma-Seminar Fragebogen 1

Wie gut kennen Sie MathePrisma?

2	oft angesehen
3	selten angesehen
1	gar nicht

Welche Erfahrungen haben Sie im Nutzen von Standardanwendungen (Word, Excel, ...)?

6	oft verwendet
0	selten verwendet
0	gar keine

Welche Erfahrungen haben Sie im Umgang mit dem Internet?

6	Suchen von Informationen (z.B. mit Google)
5	Download von Dateien
5	Online-Bestellung (z.B. über amazon)
0	gar keine

Welche Erfahrungen haben Sie im Erstellen von Internetseiten?

0	Kenntnisse in HTML
1	Verwenden von HTML-Editoren
0	Erstellen von JavaScript
5	gar keine

Welche Designüberlegungen sind Ihnen bei Internetseiten wichtig?

Welche Erfahrungen haben Sie im Bearbeiten von Dateien?

6	Umbenennen
6	Kopieren
2	Komprimieren (z.B. mit WinZip)
0	gar keine

Welche Erfahrungen haben Sie im Bearbeiten von Bildern?

6	Ändern der Größe
4	Ändern des Dateiformats
0	gar keine

Welche Erfahrungen haben Sie im Verwenden von Latex-Notation?

0	oft verwendet
0	selten verwendet
6	gar keine

Welche Betriebssysteme haben Sie bereits benutzt?

6	Windows
2	Linux
0	sonstige:
0	gar keine

Welche Programmiersprachen beherrschen Sie?

0	C / C++
0	Java
2	sonstige:
4	gar keine

Welche Didaktikveranstaltungen haben Sie bereits besucht?

0	Didaktik der Algebra
0	Didaktik der Analysis
1	Didaktik der Arithmetik
0	Didaktik der Geometrie
0	Didaktik der Stochastik
0	Didaktik der Zahlenbereiche
5	sonstige:
0	gar keine

Welche praktischen Erfahrungen haben Sie im Entwickeln von Unterrichtseinheiten?

6	Praktikum besucht:
0	eLearning erstellt
6	Nachhilfe gegeben
0	gar keine

Beschreiben Sie stichpunktartig, wie Sie eine Unterrichtseinheit konzipieren würden?

(Name oder Geburtsdatum der Mutter)

MathePrisma-Seminar Fragebogen 2

Wie beurteilen Sie die Benutzerfreundlichkeit von teachTool?	[0] ++	[6] +	[2] +−	[0] −	[0] −−
Wie beurteilen Sie die Übersichtlichkeit von teachTool?	[3] ++	[2] +	[2] +−	[1] −	[0] −−
Wie beurteilen Sie die Benutzerfreundlichkeit des Slideshow-Editors?	[2] ++	[2] +	[3] +−	[1] −	[0] −−
Wie beurteilen Sie die Benutzerfreundlichkeit des Math-Editors?	[0] ++	[3] +	[3] +−	[2] −	[0] −−
Wie beurteilen Sie die Benutzerfreundlichkeit des MultipleChoice-Editors?	[1] ++	[5] +	[2] +−	[0] −	[0] −−
Wie klar kommt die Trennung zwischen Struktur und Inhalt auf der Oberfläche von teachTool heraus?	[3] ++	[3] +	[1] +−	[0] −	[0] −−

Welche multimedialen Elemente sind Ihnen wichtig?

- [6] Diashow
- [8] Frage-Antwort-Aufgaben
- [8] MultipleChoice
- [2] Film
- [3] MouseOver
- [0] sonstige:

Welche multimedialen Elemente
fehlen bei teachTool?

MathePrisma-Module werden strukturiert nach dem Grad der Interaktion.
Wie wichtig erscheinen Ihnen
weitere Strukturierungselemente
wie Bemerkung, Beispiel, ...?

[2] ++ [3] + [0] +− [3] − [0] −−

Erscheint Ihnen die Verwendung
von Computer-Algebra-Systemen
als wichtig?

[3] Nein
[5] Ja,
z.B. bei:

Wie leicht fiel Ihnen bei der
bisherigen Vorbereitung die
Stoffsequenzierung
(ohne Computer / teachTool)?

[1] ++ [6] + [0] +− [1] − [0] −−

Wozu würden Sie gerne eine
Hilfefunktion haben?

[3] Slideshow-Editor
[3] Math-Editor
[1] MultipleChoice-Editor
[3] Einfügen von Strukturelementen
[0] sonstige:

Welche (Bedienungs-)Probleme /
Verbesserungsideen haben Sie?

(Name oder Geburtsdatum der Mutter)

MathePrisma-Seminar Fragebogen 3

Wie beurteilen Sie die Benutzerfreundlichkeit von teachTool?

[0] ++ [2] + [1] +− [0] − [0] −−

Wie beurteilen Sie die Übersichtlichkeit von teachTool?

[0] ++ [3] + [0] +− [0] − [0] −−

Wie sehr war Ihnen Ihr mathematisches Vorwissen bei der Bedienung von teachTool behilflich?

[0] ++ [0] + [1] +− [2] − [0] −−

Welche Funktionalitäten fehlen bei teachTool?

Welche Bedienungsprobleme / technischen Probleme hatten Sie?

Hat sich Ihre Einstellung zur Unterrichtsplanung geändert?

[1] Nein
[2] Ja,
z.B. bei (der Strukturierung, Berücksichtigung von Handlungsorientierung):

Wie wichtig erscheint Ihnen der 'Didactics Check'?	[1] ++	[0] +	[1] +−	[0] −	[0] −−

Haben Sie den 'Didactics Check' verwendet?	[2] Ja [1] Nein

Wie gut konnte dieser Sie bei der (überarbeiteten) Stoffsequenzierung unterstützen?	[1] ++	[0] +	[0] +−	[1] −	[0] −−
Wie bewerten Sie die zugrundeliegenden Kriterien?	[1] ++	[1] +	[0] +−	[0] −	[0] −−

Wie gut können Sie sich vorstellen teachTool als Lehrer zu verwenden ...

... um Module zu erstellen?	[2] ++	[0] +	[0] +−	[0] −	[0] −−
... um Module auf Ihre Bedürfnisse im Unterricht anzupassen?	[3] ++	[0] +	[0] +−	[0] −	[0] −−
Wie bewerten Sie das MathePrisma-Seminar?	[1] ++	[2] +	[0] +−	[0] −	[0] −−

Welche Verbessungsvorschläge, Anregungen oder Bemerkungen haben Sie zum MathePrisma-Seminar?

Name (optional, für Nachfragen): ☐

Vorkenntnisse

Wie viel Erfahrung haben Sie im Nutzen von Textverarbeitungsprogrammen (z.B. Word)?

10; 5	viel
1; 3	wenig
0; 0	keine

Welche Erfahrungen haben Sie im Umgang mit dem Internet?

12; 8	Suchen von Informationen (z.B. mit Google)
12; 6	Download von Dateien
11; 7	Online-Bestellungen (z.B. über amazon)
0; 0	keine

Welche Erfahrungen haben Sie im Erstellen von Internetseiten?

2; 2	Erstellen von JavaScript
6; 2	Kenntnisse in HTML
3; 2	Verwenden von HTML-Editoren (z.B. Frontpage, Dreamweaver)
5; 6	keine

Wie viel Erfahrung haben Sie im Verwenden von LaTeX?

4; 0	viel
2; 5	wenig
6; 3	keine

Welche Programmiersprachen beherrschen Sie?

7; 1	C / C++
6; 2	Java
7; 4	sonstige: ☐
2; 4	keine

Technik

Wie schätzen Sie Ihre Eingewöhnungszeit für teachTool ein (im Vergleich zu Standardanwendungen wie Word, Excel, ...)?

0; 0	sehr hoch
2; 1	hoch
2; 2	mittel
8; 5	niedrig
1; 0	sehr niedrig

Wie beurteilen Sie die Benutzerfreundlichkeit von teachTool?

4; 0	++
5; 7	+
3; 1	+−
0; 0	−
0; 0	−−

Wie beurteilen Sie die Übersichtlichkeit von teachTool?

4; 1	++
8; 3	+
0; 4	+−
0; 0	−
0; 0	−−

Wie beurteilen Sie die Benutzerfreundlichkeit des Slideshow-Editors?

1; 0	++
5; 0	+
3; 3	+−
1; 4	−
0; 0	−−

Wie beurteilen Sie die Benutzerfreundlichkeit des Math-Editors?

2; 0	++
3; 4	+
6; 2	+−
1; 2	−
0; 1	−−

Wie beurteilen Sie die Benutzerfreundlichkeit des Multiplechoice-Editors?

2; 1	++
4; 1	+
2; 3	+−
0; 0	−
0; 0	−−

Wie klar kommt die Trennung zwischen Struktur und Inhalt auf der Oberfläche von teachTool heraus?

2; 1	++
7; 2	+
2; 3	+−
1; 0	−
0; 0	−−

Entsprechen die HTML-Seiten dem, was Sie von Ihren Eingaben in teachTool erwarten?

2; 3	++
6; 3	+
2; 2	+−
2; 0	−
0; 0	−−

Welche interaktiven Elemente sind Ihnen wichtig?

8; 4	Diashow
5; 5	Texteingabe-Antworten
6; 3	Multiplechoice
6; 5	Film
2; 1	MouseOver
0; 0	Puzzle
0; 1	Hotspot
8; 7	Applet
0; 0	sonstige:

MathePrisma-Module werden strukturiert nach dem Grad der Interaktion in verbindende/erläuternde Texte, Definitionen/(Merk-)Sätze und Interaktionen.
Wie wichtig erscheint es Ihnen, weitere inhaltlich orientierte Strukturierungselemente wie Bemerkung oder Beispiel hinzuzufügen?

2; 3	++
3; 2	+
2; 1	+−
1; 0	−
1; 1	−−

Welche Strukturierungselemente fehlen bei teachTool?

Welche Funktionalitäten (z.B. stilistische bzw. interaktive Elemente, Design-Optionen) fehlen bei teachTool?

Erscheint Ihnen allgemein die Verwendung von Computer-Algebra-Systemen im Rahmen von E-Learning-Modulen als wichtig?

2; 4	Nein
3; 2	Ja, z.B. bei:

Wozu würden Sie gerne eine Erweiterung der Hilfefunktion haben?

1; 3	Slideshow-Editor
4; 3	Math-Editor
3; 0	Einfügen von Strukturelementen
0; 0	sonstige:

Welche (Bedienungs-)Probleme / technischen Probleme haben Sie bei teachTool?

Welche (Bedienungs-)Probleme / technischen Probleme haben Sie bei den fertigen Lerneinheiten (im Browser)? Welcher Browser wird dabei verwendet?

Welche Verbesserungsideen haben Sie?

Didaktische Benutzerführung

Wie wichtig erscheint Ihnen der 'Didactics Check'?

3; 0	++
2; 0	+
3; 1	+−
2; 3	−
1; 0	−−

Haben Sie den 'Didactics Check' verwendet?

4; 1	Ja
8; 6	Nein

Wie gut konnte dieser Sie bei der Stoffsequenzierung unterstützen?

1; 0	++
3; 0	+
0; 0	+−
1; 1	−
1; 0	−−

Wie bewerten Sie die zugrundeliegenden Kriterien?

0; 0	++
3; 0	+
2; 1	+−
0; 0	−
0; 0	−−

Welche Änderungen haben Sie aufgrund der Hinweise des 'Didactics Checks' vorgenommen?

1; 0	Entfernen von Kapiteln
1; 0	Hinzufügen von Kapiteln
2; 0	Aufteilen von Kapiteln
3; 0	Entfernen von Seiten
1; 0	Hinzufügen von Seiten
3; 1	Aufteilen von Seiten
2; 0	Entfernen von Abschnitten
0; 0	Hinzufügen von Abschnitten
2; 0	Aufteilen von Abschnitten
5; 0	Umstrukturierung von Seiten
5; 0	Verschieben von Informationen auf Nebenpfadseiten
4; 0	Kürzen von Texten
4; 0	Hinzufügen von Bildern
3; 0	Hinzufügen von Interaktionen
0; 0	sonstiges:

Tabellenverzeichnis

Abbildungsverzeichnis

Literaturverzeichnis

[ADL] ADL - Advanced Distributed Learning. *Advanced Distributed Learning - Home*. http://www.adlnet.org/

[Adoa] Adobe Systems Incorporated. *Macromedia - Authorware 7* . http://www.adobe.com/products/authorware/

[Adob] Adobe Systems Incorporated. *Macromedia - Director MX 2004*. http://www.adobe.com/products/director/

[Adoc] Adobe Systems Incorporated. *Macromedia - Flash Professional 8*. http://www.adobe.com/products/flash/flashpro/

[Arb] Arbeitsgruppen Didaktik und Angewandte Informatik im Fachbereich C / Mathematik der Bergischen Universität Wuppertal. *MathePrisma*. http://www.matheprisma.de/

[Bal00] Balzert, H.: *Lehrbuch der Software-Technik*. Bd. 1: *Software-Entwicklung*. 2. Aufl. Heidelberg : Spektrum Akademischer Verlag, 2000

[BB96] Ballin, D. ; Brater, M.: *Handlungsorientiert lernen mit Multimedia*. 1. Aufl. Nürnberg : BW Bildung und Wissen Verlag und Software GmbH, 1996

[BFB] Boltze, M. ; Friedrich, B. ; Bastian, M. (Verantwortliche). *OptiV*. http://www.optiv.de/

[BKV03] Blankenagel, K. ; Krivsky-Velten, S.: MathePrisma. In: Schweizerische Koordinationsstelle für Bildungsforschung SKBF (Hrsg.): *Lehrerinnen- und Lehrerbildung Kanton und Universität Bern (LLB) Schweizerische Gesellschaft für Bildungsforschung (SGBF)*. Aarau : Schweizerische Gesellschaft für Lehrerinnen- und Lehrerbildung (SGL), 2003 (Beiträge des Jahreskongresses Schule und Familie - Perspektiven einer Differenz 9)

[BKV04] Blankenagel, K. ; Krivsky-Velten, S.: Neue Zugänge zur Mathematik und ihrer Didaktik durch multimediale Autorentätigkeit. In: Heinze, A.

(Hrsg.) ; KUNTZE, S. (Hrsg.): *Beiträge zum Mathematikunterricht: Vorträge auf der 38. Tagung für Didaktik der Mathematik*. Hildesheim, Berlin : Verlag Franzbecker, 2004, S. 105–108

[Boe02] BOESKEN, G.: Lesen am Computer - Mehrwert oder mehr Verwirrung? Untersuchungen zur „Konkurrenz“ zwischen Buch und Hypertext. In: BRAUNGART, G. (Hrsg.) ; EIBL, K. (Hrsg.) ; JANNIDIS, F. (Hrsg.): *Jahrbuch für Computerphilologie 4*. Paderborn : mentis Verlag, 2002, S. 85–114

[CH02] CHAPMAN, B. ; HALL, B.: Authoring Tool Strategies. (2002)

[Chi02] CHINA, R. (DLC-STUDIE): E-Learning-Produkte im Vergleich. In: *Personalwirtschaft, Sonderheft E-Learning (11/2002)*. 2002, S. 38–48

[DEL] DELFI SOFTWARE. *LERSUS :: E-Learning Autorenwerkzeug / Produkt & Lösungen / Autorenwerkzeug*. http://www.lersus.de/

[Des] DESIGN SCIENCE. *MathType: Download TeXaide* . https://www.dessci.com/en/products/texaide/

[Dil] DILTHEY, A. *AboutWebDesign.de - Schriften/Textgestaltung* . http://www.aboutwebdesign.de/awd/content/970612510.shtml

[Ern05] ERNST, A.: *Konstruktivistisch orientierte Aufbereitung mathematikdidaktischer Inhalte für Hypermedia*. 1. Aufl. Hildesheim, Berlin : Verlag Franzbecker, 2005

[FGK00] FROMMER, A. ; GROSSER, B. ; KRIVSKY, S.: MathePrisma – Erfahrungen mit einem Multimedia-Projekt zur Mathematik für die Schule. In: BLANKENAGEL, J. (Hrsg.) ; SPIEGEL, W. (Hrsg.): *Mathematikdidaktik aus Begeisterung für die Mathematik*. Stuttgart : Ernst Klett Schulbuchverlag, 2000, S. 73–82

[FKV04] FROMMER, A. ; KRIVSKY-VELTEN, S.: MathePrisma. In: BRAKE, C. (Hrsg.) ; TOPPER, M. (Hrsg.) ; WEDEKIND, J. (Hrsg.): *Der MEDIDA-PRIX - Nachhaltigkeit durch Wettbewerb*. Münster : Waxmann Verlag, 2004

[Fre03] FREIBICHLER, H.: Entwicklungswerkzeuge für E-Learning auswählen. In: HOHENSTEIN, A. (Hrsg.) ; WILBERS, K. (Hrsg.): *Handbuch E-Learning*. Köln : Fachverlag Deutscher Wirtschaftsdienst, 2003

[Glü04] GLÜCKSELIG, S. *Standards im e-learning Umfeld: SCORM* . http://www.gungfu.de/studium/e-learning_scorm/vortrag.html. 2004

[Gra05] GRABMEIER, J.: Students learn better when the Numbers don't talk and dance. In: TOEPELL, M. (Hrsg.): *Mitteilungen der Gesellschaft für Didaktik der Mathematik, Nr. 81*. Leipzig : Merkur GmbH Leipzig, 2005, S. 117–119

[GSW90] Glaser, H. ; Scheid, H. ; Wellstein, H.: *SIGMA-Grundkurs Stochastik*. Stuttgart : Klett-Verlag, 1990

[Hal] Hall, B. *Authoring Systems Buyer Guide 2001*. Brandon & Hall

[Hey00] Heyder, F.: *Autorensysteme - moderne Werkzeuge für die Schule*. Bad Heilbrunn : Klinkhardt, 2000

[Hof] Hoffmann, T. *jDvi*. http://www-sfb288.math.tu-berlin.de/jdvi/home.html

[ILI] ILIAS. *ILIAS open source*. http://www.ilias.de/

[Inf01] Information Society Technologies (IST) Programme of the European Union: Learning Management Systems & Authoring Tools. (2001)

[Ker01] Kerres, M.: *Multimediale und telemediale Lernumgebungen - Konzeption und Entwicklung*. 2. Aufl. München : Oldenbourg Verlag, 2001

[Ker02] Kerkau, F.: Autorenwerkzeuge für Online-Lernangebote. In: Issing, L. (Hrsg.) ; Klimsa, P. (Hrsg.): *Information und Lernen mit Multimedia*. Weinheim : Beltz Verlagsgruppe, 2002, S. 218–226

[Kor] Kortenkamp, U. *Cinderella : Die interaktive Geometrie-Software Cinderella*. http://cinderella.de/

[Kri03] Krivsky, S.: *Multimediale Lernumgebungen in der Mathematik*. 1. Aufl. Hildesheim, Berlin : Verlag Franzbecker, 2003

[Leh] Lehrstuhl für Mathematik und ihre Didaktik der Universität Bayreuth. *http://geonext.de []*. http://geonext.uni-bayreuth.de/

[LHR99] Lynch, P. J. ; Horton, S. ; Rosdale, R. M.: *Erfolgreiches Web-Design*. München : Humboldt Taschenbuchverlag Jakobi KG, 1999

[Lin] Link & Link Software. *IDEA – Link und Link Software GmbH & Co. KG - Home*. http://www.linkundlink.de/cms/

[Mey94] Meyer, H.: *Unterrichtsmethoden*. Bd. 1: *Theorieband*. 6. Auflage. Frankfurt : Cornelsen Scriptor, 1994

[moo] moodle. *Moodle - A Free, Open Source Course Management System for Online Learning*. http://moodle.org/

[NV05] Nantel, R. ; Vipond, S.: Authoring Tool KnowledgeBase 2006: A Buyer's Guide to 100 of the Best E-Learning Content Development Applications. (2005)

[Paw04] Pawlowski, J. M.: E-Learning-Standards. In: *E-Learning in Hochschulen und Bildungszentren*. 2004, S. 454–472

[Pay02] Payone, T.: Jeder kann jetzt Autor sein. In: *Weiterbildung & Wirtschaft, Heft 9/2002*. 2002, S. 52–54

[Pet04] Peter, M. *An integrated development environment for interactive mathematical documents* . http://omnibus.uni-freiburg.de/~mp4/MPEditor/mpe.pdf. 2004

[Pos] Poskanzer, J. *ACME*. http://www.acme.com/

[REL] RELOAD. *RELOAD Project*. http://www.reload.ac.uk/

[RP89] Rode, M. ; Poirot, J.: Authoring systems-are they used? In: *Journal of Research on Computing in Education, Vol. 22, No. 2*. 1989, S. 191–198

[Ruh] Ruhr-Universität Bochum. *HotEqn* . http://www.esr.ruhr-uni-bochum.de/VCLab/software/HotEqn/HotEqn.html

[Sac90] Sacher, W.: *Computer und die Krise des Lernens*. Bad Heilbrunn : Klinkhardt, 1990

[Sch02] Schulmeister, R.: *Grundlagen hypermedialer Lernsysteme. Theorie - Didaktik - Design*. 3. Aufl. München, Wien : Oldenbourg Verlag, 2002

[Sch03] Schulmeister, R.: *Lernplattformen für das virtuelle Lernen - Evaluation und Didaktik*. München, Wien : Oldenbourg Verlag, 2003

[SS91] Schmid, A. ; Schweizer, W.: *Lambacher-Schweizer, Stochastik, Grundkurs*. Stuttgart : Klett-Verlag, 1991

[Ste] Stein, M. (Verantwortlich für die Inhalte). *MaDiN - Mathematik-Didaktik im Netz*. http://www.madin.net/

[Sum] SumTotal Systems, Inc. *ToolBook – e-Learning Software Content Authoring Tool — Quiz Maker Software — Assessment Test Software*. http://www.toolbook.com/

[Sun] Sun Microsystems, Inc. *The Java[tm] Tutorial* . http://java.sun.com/docs/books/tutorial/

[W3C] W3C. *MathML – W3C Math Home*. http://www.w3.org/Math/

[Wal01] Walther, P. *Werkzeuge für WBT & Co*. managerSeminare, Heft 2 e-Le@rning. 2001

[Wei03] Weiss, R. *„Rapid E-Learning“ senkt Kosten und halbiert Entwicklungszeit.* Macromedia Central Europe. 2003

[Wie] Wiegand, S. (Projekt Manager). *TeXnicCenter – About* . http://www.toolscenter.org/