

Diplomarbeit

*iTeach*

Ein adaptives hypermediales Lehrsystem

Cand. Inform. Axel Arne Guicking

30. Juli 2001

Betreuer:

Dipl. Inform. Christian Betz

Prof. Dr. Frank Puppe

---

---

Lehrstuhl für Informatik VI  
– Künstliche Intelligenz und Angewandte Informatik –  
Bayerische Julius-Maximilians-Universität Würzburg

---

---

## Zusammenfassung

Im Rahmen dieser Arbeit wird ein adaptives hypermediales Lehrsystem vorgestellt, das sich an bereits bestehenden und in der Praxis erfolgreich eingesetzten Systemen aus diesem Bereich orientiert. Es ist als serverseitige Anwendung in Java implementiert und basiert auf den Open-Source-Projekten Avalon und Cocoon der Apache Software Foundation. Die damit einhergehende modulare Architektur, die XML-basierte Datenhaltung sowie der Einsatz verschiedener Entwurfsmuster bei der Implementierung ermöglichen den Einsatz in verschiedenen Umgebungen und eine zukünftige leichte Erweiterbarkeit. Insbesondere der Einsatz von XML als zugrunde liegendes Datenformat geht über bisherige Ansätze hinaus und ermöglicht die vollständige Trennung von Inhalt, Logik und Layout.

Durch die Adaptivität des Systems kann auf den einzelnen Benutzer eingegangen werden, was durch ein individuelles Wissensmodell ermöglicht wird. Dieses Modell ist nicht monoton, d. h. es wird nicht nur die Zunahme des Wissens protokolliert, sondern es kann ebenso das Vergessen von Informationen anhand von Konfigurationsparametern gesteuert werden.

Die adaptive Führung und Präsentation basiert auf verschiedenen Techniken und ist in der Regel nicht-restriktiv, sondern bietet vielmehr Navigations-Vorschläge z. B. in Form von sog. Guided Tours an, der Benutzer ist dabei nicht gezwungen, dem vorgeschlagenen Pfad zu folgen. Ferner kann der Benutzer durch Einsicht in sein Wissensmodell vom System generierte Annahmen überschreiben und zurücksetzen.

## Abstract

An educational adaptive hypermedia system is presented in this work. It is oriented on existing systems of this research area which have been tested and used in different courses. It is implemented as a server-side Java application based on the open source frameworks Avalon and Cocoon, both projects of the Apache Software Foundation. The modular architecture, the XML-based data management and the use of several design patterns make it easy to use this system in different contexts and to extend its features in future. Especially the usage of XML as basic data format exceeds existing research and makes it possible to separate contents, logic and layout.

The system is able to recognize the individual user and to adapt its behavior to his needs by modeling the knowledge of each user. This knowledge model is nonmonotonic, i.e. not only the knowledge increase can be modeled but also the knowledge decay which can be controlled by certain configuration parameters.

The adaptive guidance and presentation is based on different techniques which are usually non-restrictive, meaning that the system makes suggestions for the navigational order but the user does not need to follow these recommendations. Besides, the user can change his own user model and override the assumptions made by the system.

# Inhaltsverzeichnis

<b>1</b>	<b>Einleitung</b>	<b>1</b>
<b>2</b>	<b>Computerunterstütztes Lernen</b>	<b>3</b>
2.1	Klassische oder computergestützte Ausbildung? . . . . .	4
2.2	Problematik computerbasierter Lernens . . . . .	6
2.3	Praktischer Einsatz computerbasierter Ausbildung . . . . .	8
<b>3</b>	<b>Adaptive Hypermediasysteme</b>	<b>11</b>
3.1	Adaptive Hypermedia . . . . .	12
3.2	Klassifikation von AH-Systemen . . . . .	12
3.3	Einsatzgebiete für AH . . . . .	13
3.4	Adaptionsinformationen . . . . .	15
3.5	Adaptionsziele . . . . .	16
3.6	Adaptionsmethoden und -techniken . . . . .	17
3.7	Evaluationen adaptiver Techniken . . . . .	23
<b>4</b>	<b>Adaptive Lehrsysteme in der Praxis</b>	<b>26</b>
4.1	Domänenmodell . . . . .	27
4.2	Benutzermodell . . . . .	28
4.3	Adaptionskomponente . . . . .	30
4.4	Inferenzkomponente . . . . .	30
4.5	Vergleich bestehender Lehrsysteme . . . . .	31
<b>5</b>	<b>Beispielinhalte</b>	<b>37</b>
5.1	Der Kriterienkatalog . . . . .	37
5.2	Ergebnis der Evaluation . . . . .	42
5.3	Adaptive Hypermedia . . . . .	43
<b>6</b>	<b>Adaptivität in iTeach</b>	<b>45</b>
6.1	Das Domänenmodell . . . . .	45
6.2	Das Benutzermodell . . . . .	50
6.3	Adaptive Techniken . . . . .	52
6.4	Die Inferenzkomponente . . . . .	55

---

<b>7</b>	<b>Einsatzmöglichkeiten von iTeach</b>	<b>62</b>
7.1	iTeach als eigenständige Webapplikation . . . . .	62
7.2	iTeach im Informationssystem . . . . .	68
7.3	Java Online Praktikum . . . . .	69
<b>8</b>	<b>Implementierung von iTeach</b>	<b>71</b>
8.1	Entscheidungen zur Implementierung . . . . .	71
8.2	Verwendete Frameworks . . . . .	73
8.3	Klassen und Pakete von iTeach . . . . .	76
<b>9</b>	<b>Ausblick</b>	<b>83</b>
9.1	Erweiterungsmöglichkeiten . . . . .	85
9.2	Abschließender Ausblick . . . . .	87
<b>A</b>	<b>Evaluation der Java-Kurse</b>	<b>90</b>
A.1	Allgemeines . . . . .	90
A.2	Inhalt . . . . .	91
A.3	Präsentation . . . . .	92
A.4	Bedienung . . . . .	94
A.5	Gesamtbeurteilung . . . . .	95
<b>B</b>	<b>Dateien und Verzeichnisse</b>	<b>97</b>
B.1	Verzeichnisstruktur . . . . .	97
B.2	Formate der wichtigsten XML-Dateien . . . . .	98
<b>C</b>	<b>Installation und Konfiguration</b>	<b>100</b>
	<b>Literaturverzeichnis</b>	<b>102</b>
	<b>Verzeichnis der Internetadressen</b>	<b>107</b>

# Kapitel 1

## Einleitung

*„Im Internet kann man alles finden, nur keine Weisheit.“*

*Jostein Gaarder*

Im Jahre 1989 wurde die Hypertext Markup Language (HTML) als Auszeichnungssprache für Dokumente von Tim Berners-Lee entwickelt, die auf der Standard Generalized Markup Language (SGML) basiert [2]. Seit dem explosionsartigen Anstieg der Nutzer des World Wide Web (WWW) durch Einführung des ersten Webbrowsers XMosaic im Jahre 1993 nimmt der Umfang der verfügbaren Informationen auch heute noch rasant zu<sup>1</sup>.

Die mit dieser Entwicklung einhergehende Schwierigkeit der Daten-Organisation besteht vornehmlich darin, die dezentralen Daten überhaupt erst einmal zu finden, die in den Dokumenten enthaltenen Informationen zu extrahieren und schließlich diese untereinander austauschbar zu machen – sei es zwischen Endbenutzer und Suchmaschine oder zwischen verschiedenen Institutionen; darauf spielt wohl letztlich auch obiges Statement Gaarders an.

Seit 1996 hat die Extensible Markup Language (XML) des World Wide Web Consortiums (W3C) eine vergleichbar stürmische – wenn auch „stillere“ – Entwicklung erfahren [2]. XML basiert wie HTML auf SGML; während HTML aber nur eine „Ausprägung“ darstellt, können mit XML beliebig viele Markup-Sprachen definiert werden. Durch den verfolgten Generic Markup-Ansatz ist es möglich, anhand der Metadaten (den Elementnamen) die *Semantik* des Dokuments zu analysieren. Von Vorteil gegenüber anderen Textauszeichnungssprachen wie  $\text{T}_{\text{E}}\text{X}$  bzw.  $\text{L}^{\text{A}}\text{T}_{\text{E}}\text{X}$  [23] sind die für Computer und Menschen gleichermaßen leicht verständliche Form sowie die weitergehenden Verarbeitungsmöglichkeiten [2]. Durch Analyse der Elementnamen ist es prinzipiell möglich, Suchanfragen durch Einbeziehung dieser Metadaten wesentlich aussagekräftiger formulieren und die Treffermenge reduzieren zu können. Auch wenn mit einer Verdrängung von HTML durch XML als Standardauszeichnungssprache des WWW nicht zu rechnen ist (und dies vermutlich auch nicht unbedingt sinnvoll ist), liefert XML die solide Basis für ein wesentlich besseres Datenmanagement im WWW.

---

<sup>1</sup><http://www.cyveillance.com/us/newsroom/pressr/000710.asp>

Durch die zunehmende Nutzung des WWW durch Privatpersonen eröffnen sich viele neue Anwendungsbereiche; neben dem reinen Informationsaustausch (der nach wie vor zu großen Teilen über die älteren Internet-Dienste E-Mail und Net News abgewickelt wird) sind es vor allem die kommerzielle Nutzung (E-Commerce) und die Ausbildung im weitesten Sinne, die sowohl für private Nutzer als auch für Unternehmen in Industrie und Wirtschaft von Interesse sind.

Die besonders in den letzten ein bis zwei Jahren in den Medien und der Politik allgegenwärtigen Schlagworte „Lernen via Internet“ und „E-Learning“ machen deutlich, welcher wichtiger Zweig dies ist [24], [48], [28]. Die Vorteile von Ausbildung via Internet liegen auf der Hand: Neben der Möglichkeit, mehr und unterschiedliche Zielgruppen erreichen und ansprechen zu können, ist vor allem die Unabhängigkeit von Ort und Zeit die Ursache für das öffentliche Interesse – in unserer zunehmend mobilen und globalisierten Gesellschaft zwei wichtige Faktoren.

Auf der Learntec 2001 in Karlsruhe wurden neben der Ausbildung via Internet im Allgemeinen u. a. auch so genannte „Bots“ vorgestellt, die eine personalisierte Benutzerführung ermöglichen und damit eine Adaptivität an den individuellen Benutzer ermöglichen [48]. Der Vorteil von adaptiven Lehrsystemen liegt ganz allgemein darin, dass das Verhalten und (Hintergrund-)Wissen des Benutzers berücksichtigt wird, um Benutzerschnittstelle und Inhalte des Systems an die jeweiligen Bedürfnisse anzupassen.

Das im Rahmen dieser Arbeit vorgestellte und implementierte System iTeach kombiniert die Ansätze von XML und adaptiven Lehrsystemen in dem Sinne, dass auf der einen Seite die zugrunde liegenden Informationseinheiten und die für eine Benutzeradaptivität erforderlichen Strukturdaten in XML gehalten werden, und auf der anderen Seite sowohl der vermittelte Lehrinhalt als auch die Hyperlinkstruktur an den individuellen Benutzer angepasst wird.

Die vorliegende Arbeit gliedert sich in zwei Teile: In den Kapiteln 2 bis 4 werden die theoretischen Hintergründe und der Bedarf an computerbasierten Lehrsystemen näher erläutert, wobei das Hauptaugenmerk auf adaptiven Lehrsystemen liegt. Im zweiten Teil (den Kapiteln 5 bis 8) wird das im Rahmen dieser Arbeit implementierte System und die damit zusammenhängenden Aufgaben beschrieben, bevor im letzten Kapitel ein Ausblick auf die zukünftige Entwicklung von computerbasierten Lehrsystemen im Allgemeinen und die Erweiterungsmöglichkeiten des implementierten Systems im Speziellen gegeben wird.

# Kapitel 2

## Computerunterstütztes Lernen

*„Computer sind nutzlos. Sie können uns nur Antwort geben.“*

*Pablo Picasso*

Durch den heute nicht mehr wegzudenkenden (und nicht nur nutzlosen) Einsatz von Computern in verschiedensten Bereichen steigt auch in der Ausbildung und Lehre der Bedarf an Computer-Unterstützung. Durch zunehmend komplexere Arbeitsabläufe und steigende Qualitätsanforderungen sowie durch die Entwicklung neuer und umfangreicher Anwendungsprogramme sind Schulungen erforderlich, die sowohl zeit-, als auch kostenintensiv sind. Aus diesem Grund wird in wirtschaftlichen Betrieben vermehrt das Konzept vom „Lernen am Arbeitsplatz“ verfolgt, für dessen Umsetzung computerbasierte Lernumgebungen erforderlich sind.

Durch die anhaltende rasante Entwicklung des Internet, insbesondere des World Wide Web (WWW), ergeben sich neue Möglichkeiten der Ausbildung. Dies betrifft nicht nur die innerbetriebliche Ausbildung, sondern auch an Hochschulen werden zunehmend Online-Kurse angeboten, die als Erweiterung von und Konkurrenz zu den klassischen Fernuniversitäten angesehen werden können. Über das WWW können solche Kurse auch der ständig steigenden Zahl von privaten Nutzern im Internet auf einfache Weise zugänglich gemacht werden, sodass interessierte Nutzer die Möglichkeit zur zeit- und ortsunabhängigen individuellen Aus- und Weiterbildung erhalten. Ferner können auf diese Weise auch neue Zielgruppen wie Alleinerziehende, Behinderte u. a. erreicht werden.

Computerunterstütztes Lernen (CUL)<sup>1</sup> hat in den letzten Jahren eine enorme Entwicklung erfahren. Blumstengel nennt dafür verschiedene Gründe [3]:

- Notwendigkeit flexibler und individualisierter Lernumgebungen
- Weiterentwicklung didaktischer Konzepte

---

<sup>1</sup>In der Literatur werden parallel dazu eine Reihe anderer Begriffe verwendet, von denen die Wichtigsten sind: *computer based training* (CBT), *computer aided/assisted learning* (CAL), *computer-aided teaching* (CAT) und *computer aided/assisted instruction* (CAI). Eine klare Trennung ist aufgrund der Ähnlichkeit der Begriffe schwierig und hier nicht erforderlich.

- Zunehmender Preisverfall von PC-Hardware
- Multimediafähigkeit bei heutigen PCs
- Gute Kompressionsverfahren zur Übermittlung von Multimedia-Daten
- Einsatz besserer Software-Entwicklungswerkzeuge
- Entwicklung des World Wide Web und zunehmende Nutzung durch Privatpersonen

An den Universitäten ist der Einsatz von CUL deutlich geringer als in der Wirtschaft, jedoch ist auch hier ein starker Anstieg in den letzten Jahren zu verzeichnen. Das Interesse am Einsatz von CUL ist sowohl bei Dozenten als auch bei Studenten sehr hoch, jedoch wird häufig der hohe Entwicklungsaufwand gescheut [3]. In Abschnitt 2.3 wird der praktische Einsatz von computerunterstützter Lehre in Wirtschaft und Hochschule näher betrachtet.

## 2.1 Klassische oder computergestützte Ausbildung?

Die klassische Ausbildung, bei der ein einzelner Lehrer in einer Frontalveranstaltung (Vorlesung, Seminar o. ä.) für mehrere Lernende zuständig ist, ist in der Regel mit mehreren Nachteilen verbunden:

- Die räumlichen und zeitlichen Rahmenbedingungen sind in der Regel ein Kompromiss zwischen allen Beteiligten.
- Das Lerntempo ist fest vorgegeben, was insbesondere bei unterschiedlicher Aufnahmefähigkeit der Lernenden, aber auch bei unterschiedlichen Vorkenntnissen zu Problemen führt.
- Je größer die Lerngruppe, desto schwieriger ist es für den Lehrer, auf individuelle Bedürfnisse und Fragen einzugehen.

Beim Lernen aus Büchern sind die genannten Nachteile zwar nicht vorhanden, jedoch gibt es kaum direkte Feedbackmöglichkeit, was für das Begreifen komplexer Sachverhalte unabdingbar ist.

Möglichkeiten, diese Missstände zu beheben, ist zum einen die Verbesserung der klassischen Ausbildung durch Aufteilung in kleinere Lerngruppen – ideal wäre hier eine 1:1-Beziehung von Lehrer zu Lerner, was aber aufgrund des Mangels an entsprechenden Fachkräften und den enormen Kosten unrealistisch ist. Zum anderen kann durch den Einsatz von computerbasiertem Unterricht versucht werden, die individuellen Anforderungen und Wünsche des Lernenden zu berücksichtigen und die Ausbildungsmöglichkeiten für weitere Zielgruppen zu verbessern.

Die positiven Erwartungen von CUL umfassen dabei die folgenden Punkte:

- Die relativ hohen Investitionen, die zur Erstellung der Lernumgebung erforderlich sind, werden durch die extrem geringen Vervielfältigungskosten kompensiert. Bei der betrieblichen Ausbildung entfallen die zusätzlichen Kosten für Trainer und Arbeitsausfall.
- Durch Einsatz verschiedener Medien wie Hypertext, Grafiken, Animationen, Videos und Tondokumente innerhalb eines Dokuments können die vermittelten Konzepte umfassend und leicht verständlich vermittelt werden. Teure Vorführgeräte und aufwändige Installationen entfallen durch den Einsatz eines PCs als universelles Präsentationsmedium.
- Das System kann individuelle Wünsche und Bedürfnisse der Lerner berücksichtigen; der Lerner ist zeitlich ungebunden und kann das Lerntempo selbst bestimmen. Räumliche Ungebundenheit wird durch den Einsatz von Laptops ebenfalls zunehmend erleichtert.

Es sind in erster Linie diese Gründe, die zu einem regelrechten Boom bei der Entwicklung von computergestützten Lehrsystemen geführt haben. Eine zunehmend stärker werdende Teilströmung ist in der Entwicklung von Trainingssystemen zu beobachten, die das Internet als Informationsplattform verwenden, sog. *web-basierte Trainingssysteme* (WBT). Hauptursachen für diese seit Mitte der 90er Jahre anhaltende Entwicklung sind in erster Linie die folgenden [32]:

- Änderungen des Kursmaterials müssen lediglich an einer Stelle vorgenommen werden, um alle Benutzer auf den neuesten Stand zu bringen. Dies ist insbesondere für Fachgebiete wie der IT-Branche selbst und z. B. der Bioinformatik hilfreich, in denen die Forschungsentwicklung besonders schnell voranschreitet.
- Das Internet bietet eine Fülle von Zusatzinformationen, die auf verhältnismäßig einfache Weise mit dem Lehrsystem verknüpft werden können.
- Der Lerner kann explorativ durch den Inhalt „surfen“; dies fördert die Motivation und das Interesse des Studenten am Lehrstoff.
- Neben dem WWW können weitere Dienste des Internet (E-Mail, Chat und News) zur Verbesserung von Teamarbeit und zum kollaborativen Problemlösen eingesetzt werden.

Ferner spielen die Tatsachen, dass keine teure Software gekauft werden muss, sondern frei verfügbare Webbrowser eingesetzt werden können und die potenziell sehr große Leserschaft eine entscheidende Rolle bei der Verbreitung webbasierter Lehrsysteme.

Auch in der nächsten Zukunft wird dieser Trend sicherlich anhalten, und durch die zunehmende automatische Adaptivität der hypertextbasierten Lehrsysteme an den individuellen Benutzer ist mit einem verstärkten praktischen Einsatz zu rechnen (vgl. Kap. 3 ab Seite 11).

Bei der Entwicklung neuer Medien gab es bereits früher immer wieder Äußerungen über Vorhersagen der revolutionären Verdrängung bestehender Medien. Beispielsweise äußerte sich Thomas Edison im Jahre 1922 zur Entwicklung der Filmtechnik (zitiert nach [3]):

“I believe that the motion picture is destined to revolutionize our educational system and that in a few years it will supplant largely, if not entirely, the use of textbooks.”

Heute geht die so genannte Substitutionsthese davon aus, dass die Ausbildung in ihrer klassischen Form durch computerbasierten Unterricht vollständig verdrängt wird. Perelman äußert sich dazu folgendermaßen (zitiert nach [3]):

“In the wake of the HL<sup>2</sup> revolution, the technology called ‘school’ and the social institution commonly thought of as ‘education’ will be as obsolete and ultimately extinct as the dinosaurs.”

Diese Vorhersage ist allerdings in der Literatur äußerst umstritten und wird sich nach Meinung vieler Autoren auch mittelfristig nicht bewahrheiten – vielmehr ist eine Koexistenz von klassischer und computerunterstützter Ausbildung zu erwarten. Grund für die bestehenden Zweifel sind in erster Linie neuartige Probleme und Gefahren, die durch den Einsatz computer- bzw. web-gestützter Ausbildungsmethoden entstehen können.

## 2.2 Problematik computerbasierten Lernens

Trotz der viel versprechenden Vorzüge computerbasierten Unterrichts birgt eine zu einseitige Ausbildung mithilfe des Computers eine Reihe von Gefahren, deren Ursachen in erster Linie in der komplexen nichtlinearen Struktur des Lehrstoffs liegen. Demzufolge muss bei der Entwicklung eines CUL- bzw. WBT-Systems das didaktische Konzept mit dieser Struktur abgestimmt werden [32].

### 2.2.1 Soziale Isolation

Die meisten heute bestehenden Lehrsysteme beschäftigen sich mit dem einzelnen, individuellen Lerner. Dadurch geht die soziale Komponente des Lernens und Lehrens, z. B.

---

<sup>2</sup>Hyperlearning (Anm. d. Verf.)

von klassischen Frontalveranstaltungen, verloren. Die Folge ist, insbesondere bei Lernern, die in Gruppen besser lernen können als allein, eine Einbuße an Lerneffektivität und -effizienz und schnellere Demotivation.

Durch die Einbindung vorhandener Kommunikationsmöglichkeiten des Internet (vor allem Chat, E-Mail und News) kann das klassische soziale Umfeld natürlich nicht ersetzt werden, stattdessen wird die Kommunikation zwischen Lernenden ermöglicht, die ort- und eventuell auch zeitversetzt an Problemen arbeiten. Durch die Entwicklung von Systemen, die auf das kooperative und kollaborative Lösen von Aufgaben und Problemen eingehen, erhält der Informations- und Erfahrungsaustausch zwischen den Studenten und zwischen Lehrer und Student mehr Gewicht.

### 2.2.2 Kognitive Überforderung

Neben der sozialen Isolation ist die Anforderung an den Lernenden, sich auf Meta-Ebene mit dem Lernstoff zu beschäftigen, deutlich höher als bei klassischer Ausbildung in Form einer Frontalveranstaltung oder beim Lernen aus Büchern.

#### Finden des optimalen Lernpfades

Lehrer und Buchautoren vermitteln den Lehrstoff in linearisierter Form; das Navigieren und das Finden einer didaktisch sinnvollen Präsentationsreihenfolge einzelner zu vermittelnder Konzepte wird somit vorher vom Lehrer übernommen, der aufgrund seines Wissens einen wesentlich besseren didaktischen inhaltlichen Aufbau erzielen kann. Demgegenüber bieten insbesondere WBT-Systeme eine große Zahl potenzieller Lernpfade durch den vermittelten Stoff.

Für den Lerner ist die Wahl des Verweises zum nächsten für das Erreichen seines Lernziels am besten geeignete Konzept ungleich schwerer: Da er die noch zu lernenden Konzepte nicht kennt, kann er nur durch Ausprobieren aller Verweise die optimale Reihenfolge im Nachhinein herausfinden. Aus diesem Grund bieten praktisch alle hypermedialen Lehrsysteme eine Linearisierung in Form von sog. *Guided Tours* an, die bei adaptiven Systemen den Wissensstand des Studenten berücksichtigen und die nächste zu lernende Lektion zur Laufzeit auswählen (vgl. dazu Abschnitt 3.6.3 auf Seite 20).

#### Der „Brockhauseffekt“

Durch die Vielzahl an Verweisen in den Dokumenten eines hypermedialen Lehrsystems besteht eine große Gefahr darin, dass der Student sich durch den Namen eines Verweises ablenken lässt und in ein praktisch unkontrolliertes Surfen verfällt (dieses Phänomen wird allgemein als der Lexikon- oder Brockhauseffekt bezeichnet). Daraus können sich leicht Orientierungsprobleme ergeben (s. nächsten Abschnitt); werden diese jedoch durch eine übersichtliche Gliederung (z. B. durch ein ständig sichtbares Inhaltsverzeichnis) minimiert, kann der Effekt durchaus auch zu positiven Ergebnissen führen, da der Student rein von seiner Neugier geleitet wird und damit die Lernmotivation deutlich

höher ist als bei einer durch die Software vorgegebenen Lernreihenfolge. Nachteilig ist jedoch, dass das ursprüngliche Lernziel des Studenten durch neue „kurzlebige“ Lernziele überlagert wird und damit in Vergessenheit geraten kann.

### **Orientierungsprobleme**

Der flexible Zugang zu den Lerninhalten wird durch die Gefahr der Desorientierung erkauft, die häufig mit dem Schlagwort „Lost in Hyperspace“ bezeichnet wird. Dieser fehlende Überblick über die Dokumentstruktur wird durch die folgenden Fragestellungen beschrieben [3]:

- Wo bin ich?
- Woher komme ich?
- Auf welchem Weg kam ich hierher?
- War ich hier schon einmal?
- Welche Alternativen habe ich von diesem Punkt aus?
- Gibt es mehr Informationen zum aktuellen Thema?
- Wie komme ich zu einem Punkt, der eine Übersicht bietet?
- Wie finde ich eine bestimmte Information wieder?

Durch die Bereitstellung von Navigations- und Orientierungshilfen können diese Probleme auf ein „verträgliches Maß“ reduziert werden, allerdings ist dabei zu berücksichtigen, die Benutzerschnittstelle möglichst intuitiv bedienbar zu halten (vgl. Kapitel 3.6.3).

## **2.3 Praktischer Einsatz computerbasierter Ausbildung**

Der Einsatz von CUL- und WBT-Systemen in der Praxis ist noch ausbaufähig, doch werden insbesondere im betrieblichen Umfeld bereits vermehrt solche Systeme eingesetzt, um die hohen Kosten für Mitarbeiter-Schulungen zu reduzieren. Der Einsatz an Hochschulen ist demgegenüber deutlich geringer, was vor allem auf den hohen Entwicklungsaufwand solcher Systeme zurückzuführen ist [3].

### 2.3.1 Betriebliche Aus- und Weiterbildung

Im betrieblichen Umfeld findet bereits eine rege Nutzung von computergestützten Lehrsystemen statt. Den Erfolg brachte – wie so oft in diesem Bereich – die Reduzierung der Kosten: gut bezahlte Seminarleiter werden zum Großteil durch die Lernsoftware ersetzt und durch den Einsatz von Lehrsystemen direkt am Arbeitsplatz können die durch den Arbeitsausfall entstehenden Kosten reduziert werden.

Die eingesetzten Lernsysteme lassen sich in die folgenden zwei Richtungen aufteilen: als Komponenten von Anwendungsprogrammen dienen sie zur Schulung der effizienteren Nutzung dieser Programme (z. B. Textverarbeitungs- und Tabellenkalkulationssoftware). Als eigenständige Systeme fördern sie das Wissen und die Selbsteinschätzung der Mitarbeiter im Arbeitsumfeld. Die Mehrzahl der eingesetzten CUL-Systeme bezieht sich allerdings auf den Einzel-Lerner, und nach Blumstengel ist der didaktische Hintergrund oft recht spärlich [3].

### 2.3.2 Die „Virtuelle Hochschule“

Zur Förderung des computergestützten Lernens an Hochschulen bieten inzwischen nahezu alle Bundesländer Konzepte für „virtuelle Universitäten“ an. Obwohl auch die klassische Fernuniversität Hagen<sup>3</sup> ihre Angebote vermehrt über das Internet verfügbar macht und auch diesem Verbreitungsmedium anpasst, geht das Konzept über das von Fernuniversitäten hinaus: im Vordergrund steht der vermehrte Einsatz neuer Medien sowie die damit mögliche Präsentation von Multimedia-Inhalten, der modulare Aufbau der Kurse aus kleinen Lerneinheiten und das Ansprechen weiterer Zielgruppen [24]. Als Beispiel sei hier die „Virtuelle Hochschule Bayern“ (VHB)<sup>4</sup> herausgegriffen und etwas ausführlicher dargestellt.

Angeboten werden die Kurse von allen bayerischen Universitäten und Fachhochschulen gemeinsam unter der Koordination eines Verbundinstituts, dem alle bayerischen Hochschulen angehören. Durch diese Organisation können bereits bestehende Infrastrukturen (Kursinhalte, Netzzugang u. ä.) leichter genutzt werden. Die Verteilung der Kurse auf mehrere Universitäten sorgt ferner für eine vereinfachte Aktualisierung und Wartung der Kursmaterialien und damit die Sicherung des Qualitätsstandards [50].

Das Kursangebot baut auf über 350 Projekten auf, die an bayerischen Hochschulen im Bereich virtueller Lehre entwickelt wurden [50]. Die Kurse sind in Fachgebiete, sog. „Schools“ gegliedert. Die angebotenen Fachgebiete umfassen z. Zt. Informatik, Ingenieur- und Wirtschaftswissenschaften, sowie Medizin. Später sollen weitere Fachgebiete folgen.

Die Entwicklung von Aufzeichnungsmöglichkeiten (Videoaufnahmen, Whiteboards etc.) von Veranstaltungen ermöglicht, dass ganze Vorlesungen aufgezeichnet werden können. Ein solches System, mit dem Veranstaltungen aufgezeichnet werden können,

---

<sup>3</sup><http://www.fernuni-hagen.de>

<sup>4</sup><http://www.vhb.org>

ist „Authoring on the fly“ (AOF), das an der Universität Freiburg entwickelt wird<sup>5</sup> [35]. Dieses System wird bereits an verschiedenen Universitäten in unterschiedlichsten Fachbereichen eingesetzt, u. a. auch an der „Virtuellen Hochschule Oberrhein“ (VIROR)<sup>6</sup>. Ein System mit ähnlichem Hintergrund aber etwas anderem Schwerpunkt ist „E-Kreide“<sup>7</sup>, das von einem Team um Raúl Rojas an der FU Berlin entwickelt und bereits in der Praxis erfolgreich eingesetzt wurde [40], [46]. Dieses System kombiniert die Funktionen einer althergebrachten Kreidetafel (mit ihrem pädagogischen Hauptvorteil – dem gemäßigten Unterrichtstempo), Multimedia und Internet. Die Tafel besteht aus einem berührungsempfindlichen Plasmabildschirm, auf dem mit einem Stift geschrieben werden kann; außerdem können Bilder und Applets eingebunden und mathematische Ausdrücke ausgewertet werden. Das Tafelbild wird von einem Steuerungsprogramm zusammen mit Audio- und Videodaten des Dozenten und einer Hörsaalansicht im Internet zur Verfügung gestellt. Die vollständigen Unterlagen können auch gespeichert werden, sodass E-Kreide einen viel versprechenden Ansatz sowohl für zeitabhängiges als auch zeitunabhängiges Teleteaching darstellt [40].

In Zukunft ist mit weiter sinkenden Kosten für hohe, d. h. für Echtzeitvideoübertragungen geeignete Übertragungsraten von Internetdaten zu rechnen, sodass auch vermehrt Privatpersonen solche Systeme nutzen können – zur Zeit sind jedoch lediglich Standleitungen (und mit Einschränkungen auch ADSL-Modems) in der Lage, einen vernünftigen Einsatz zu gewährleisten. In Zukunft ist auch aus diesem technischen Grund mit einer zunehmenden Entwicklung des zeit- und ortsunabhängigen Fernunterrichts zu rechnen.

---

<sup>5</sup><http://ad.informatik.uni-freiburg.de/mmgroup/aof/index.html.de>

<sup>6</sup><http://www.viror.de>

<sup>7</sup><http://www.e-kreide.de>

# Kapitel 3

## Adaptive Hypermediasysteme

*„Multimedia begann, als das erste Klavier ins Stummfilmkino geschoben wurde.“*

*Rolf Schulmeister*

Wie bereits im vorigen Kapitel beschrieben, ist der Einsatz von CUL- bzw. WBT-Systemen im Alltag zwar prinzipiell möglich, doch gerade im Bereich von Hochschulen noch nicht weit verbreitet. Ein wichtiger Aspekt in diesem Zusammenhang ist die mangelnde Fähigkeit der Systeme, auf individuelle Benutzer und deren Wünsche, Präferenzen und Ziele einzugehen. Eine solche automatische Adaption dient zum einen dazu, die Motivation zu fördern (es werden nur für ihn momentan interessante Aspekte erklärt) und zum anderen, um die Gefahr der kognitiven Überlastung (s. Abschnitt 2.2.2) zu minimieren.

Bevor die in der Praxis bereits in einigen adaptiven Lehrsystemen eingesetzten und im Rahmen dieser Arbeit implementierten Mechanismen erläutert werden, sollen die bereits stillschweigend in den vorigen Kapiteln verwendeten Begriffe Multimedia, Hypertext und Hypermedia kurz vorgestellt werden und in Verbindung zu adaptiven Hypermediasystemen gebracht werden.

*Multimedia* beschreibt das Zusammenspiel verschiedener Medien in einem Dokument, MS Encarta 99 Enzyklopädie definiert ihn beispielsweise so [19]:

*„Multimedia, die Kombination von vielen verschiedenen Medien. In erster Linie sind mit den Medien Töne, Bilder, Trick- und Videofilme gemeint. In der Computerwelt bedeutet Multimedia eine Vorstufe von Hypermedia, bei der Multimedia-Elemente über Hypertext-Methoden miteinander und mit zusätzlichen Informationen verbunden werden.“*

Der Begriff *Hypertext* wurde 1965 von Ted Nelson geprägt, basiert aber auf der Idee von Vannevar Bush (dem wissenschaftlichen Berater des damaligen US-Präsidenten Roosevelt) aus dem Jahre 1945, ein Informationssystem in Anlehnung an die assoziative Funktionsweise des menschlichen Gehirns zu schaffen [12], [44]. Hypertext wird

dementsprechend als nicht-linearer Text verstanden [3]. Durch Querverweise (Hyperlinks) zwischen den einzelnen Informationseinheiten bildet ein Hypertextsystem ein Netzwerk, durch das der Benutzer mithilfe der Hyperlinks in einer von ihm selbst gewählten Reihenfolge navigieren kann.

*Hypermedia* bildet, wie bereits aus obiger Definition von MS Encarta hervorgeht, eine Kombination von Hypertext und Multimedia [3], die insbesondere durch die Entwicklung des WWW an Popularität gewann.

### 3.1 Adaptive Hypermedia

Bei Hypertext- und Hypermediasystemen sind unabhängig vom Einsatzgebiet die Dokumente in der Regel entweder vollkommen statisch, oder die Inhalte werden dynamisch generiert (z. B. aus externen Datenquellen). Die Folge davon ist, dass für jeden Benutzer dieselben Informationen in der gleichen Reihenfolge und der gleichen Detailtiefe präsentiert werden. Da die Nutzer teilweise aber sehr unterschiedlicher Natur sind (in Bezug auf Vorwissen, Ansprüchen an das System und die verfolgten Ziele), sind die in Abschnitt 2.2.2 vorgestellten Probleme bezüglich kognitiver Überlastung des Benutzers vorprogrammiert. Abhilfe kann in erster Linie die Adaptivität der Systeme an den individuellen Benutzer schaffen. Der seit einigen Jahren etablierte englische Begriff „Adaptive Hypermedia“ (AH) wird von Brusilovsky folgendermaßen erklärt [5]:

“by adaptive hypermedia systems we mean all hypertext and hypermedia systems which reflect some features of the user in the user model and apply this model to adapt various visible aspects of the system to the user.”

Ein solches Benutzermodell umfasst die Ziele, Präferenzen und den aktuellen Wissensstand des Benutzers und dient als Grundlage für das Verhalten des Systems. Mit dem Verhalten ist in diesem Fall die Anpassung des Inhalts („content-level adaptation“) sowie der Linkstruktur („link-level adaptation“) gemeint. Im Laufe der letzten Jahre und mit steigendem Forschungsinteresse auf dem Gebiet der adaptiven Hypermediasysteme haben sich einige Methoden als sinnvoll und praxistauglich erwiesen; diese Techniken werden in Abschnitt 3.6 vorgestellt. Zunächst sollen jedoch weitere Charakteristika für eine allgemeinere Klassifizierung dieser Systeme vorgestellt werden.

### 3.2 Klassifikation von AH-Systemen

Nach Brusilovsky können adaptive Hypermediasysteme anhand der folgenden vier Dimensionen charakterisiert werden:

#### **Einsatzgebiete**

Es existieren verschiedene Anwendungsbereiche, in denen der Einsatz von adaptiven Hypermediasystemen auftretende Probleme reduzieren kann. Die einzelnen Bereiche werden im folgenden Abschnitt vorgestellt.

**Adaptionsinformationen: Woran wird angepasst**

An welche Eigenschaften des Benutzers kann das System sein Verhalten anpassen? Diese Eigenschaften werden in Abschnitt 3.4 kurz erläutert.

**Adaptionsziele: Warum wird angepasst**

Welche Ziele werden durch eine Adaption verfolgt und welche Probleme der Benutzer können gelöst oder zumindest verringert werden? (vgl. Abschnitt 3.5).

**Adaptionsmethoden: Was wird angepasst**

Welche Teile des Systems können unterschiedlich präsentiert werden? Die möglichen Verfahren und Techniken werden schließlich in Abschnitt 3.6 vorgestellt.

Im folgenden werden diese einzelnen Punkte näher betrachtet; das Hauptaugenmerk liegt dabei auf dem letzten Punkt, den Methoden und Techniken zur Adaption des Systems.

## 3.3 Einsatzgebiete für AH

Die wichtigsten Anwendungsgebiete für adaptive Hypermediasysteme sind neben der Ausbildung vor allem Online-Informations- und Hilfesysteme und Information Retrieval. Brusilovsky nennt noch weitere Bereiche, die hier jedoch nicht näher betrachtet werden sollen [5]. Die Grenzen zwischen den einzelnen Bereichen sind häufig verwaschen; demzufolge existieren Systeme, die mehreren Gebieten zugeordnet werden können [5].

### 3.3.1 Ausbildung

Aus- und Weiterbildung (von Erwachsenen) ist das größte Anwendungs- und Forschungsgebiet für adaptive Hypermediasysteme [5]. Das Ziel des Benutzers ist in der Regel klar definiert als das Lernen des gesamten oder eines Teils des Kursinhalts. Grundlage für die Adaptivität in Lehrsystemen ist das Wissen des Lerners über das behandelte Wissensgebiet.

Je nach vorhandenem Wissen des Lerners kann eine Kursseite für einen Anfänger unverständlich, gleichzeitig aber für einen Fortgeschrittenen, der sein Wissen auffrischen möchte, langweilig sein. Durch adaptive Methoden kann sowohl der Inhalt unterschiedlich präsentiert werden wie auch der vorgeschlagene Lernpfad durch das Kursmaterial an den Kenntnisstand des Benutzers angepasst werden, was insbesondere für Neulinge wichtig ist, um das „Lost in Hyperspace“-Problem zu minimieren (vgl. Abschnitt 2.2.2).

### 3.3.2 Online-Informations- und Hilfesysteme

Im Unterschied zu Lehrsystemen bieten diese Systeme keinen systematischen Einstieg sondern sind vielmehr Referenzsysteme, wie elektronische Lexika und Online-

Dokumentationen. Die Datenmenge kann je nach Domäne zwischen kleinen, überschaubaren und großen, komplexen Themengebieten variieren.

Auch bei Online-Informationssystemen besteht das Hauptproblem in der Unterstützung sehr unterschiedlicher Benutzer (vor allem in Bezug auf ihr Hintergrundwissen über die Domäne). Dies wird durch die Natur solcher Systeme noch verschärft: Den Benutzer interessiert in der Regel nur ein Teil der Information zu einem bestimmten Konzept, und das System soll diese Information möglichst schnell zur Verfügung stellen. Ferner muss das System zur Eingrenzung großer Suchbereiche und zur Navigationsunterstützung das Ziel des Benutzers kennen.

Online-Hilfesysteme können als Spezialfall von Informationssystemen aufgefasst werden, die als Teil von Softwareanwendungen direkte Hilfestellungen zum Programm und dessen Bedienung liefern. Im Unterschied zu Informationssystemen ist die Domäne in Hilfesystemen meist deutlich kleiner und das Ziel des Benutzers kann häufig direkt aus dem Kontext, in dem die Hilfe aufgerufen wurde, hergeleitet werden (sog. kontext-sensitive Hilfe).

### 3.3.3 Information Retrieval (IR)

Adaptive IR-Systeme kombinieren klassische IR-Techniken mit dem hypertextbasierten Zugriff auf die einzelnen Dokumente. Eingesetzt werden dabei häufig Ähnlichkeitsvergleiche der Dokumente, um die Relevanz der Dokumente zu gewichten. Der Domänenumfang ist in der Regel enorm; beim Einsatz von IR-Systemen im WWW ist er nahezu unbegrenzt. Im folgenden wird aufgrund der Wichtigkeit das Szenario von IR-Systemen im WWW näher betrachtet.

Im Zuge der rasanten Entwicklung des World Wide Web und der täglich (ja, sogar stündlich) wachsenden Datenmenge sind Suchmaschinen wie Google<sup>1</sup> oder Altavista<sup>2</sup> schlichtweg überfordert, die „besten“ Seiten zu finden. Der Umfang des WWW betrug im Juli 2000 nach Schätzungen von Cyveillance<sup>3</sup> über 2 Mrd. Seiten bei einem Wachstum von 7 Mio. Seiten pro Tag (!). Die Zahl der von Suchmaschinen erfassten Seiten liegt jedoch deutlich darunter. Führend ist Google mit gut 700 Mio. indizierten Webseiten, durch Berücksichtigen der Links umfasst der Suchraum jedoch über 1,3 Mrd. Webseiten<sup>4</sup>.

Da die Beurteilung einer Seite nur in seltenen Ausnahmefällen vom System zuverlässig ermittelt werden kann, können die Suchmaschinen in ihrer jetzigen Form keine optimale Treffermenge liefern. Um diese Problematik zu verdeutlichen, sei hier exemplarisch die Suche nach den Stichworten „Hawaii AND volcano“ angeführt; ein Tourist möchte sich z. B. darüber informieren, ob Hawaii in Bezug auf Vulkanausbrüche ein gefährliches Pflaster ist und ob er dementsprechend seinen nächsten Urlaub lieber auf den Seychellen verbringen sollte. Ein Extrem-Fotograf möchte mit seiner Anfrage hingegen

---

<sup>1</sup><http://www.google.com>

<sup>2</sup><http://www.altavista.com>

<sup>3</sup><http://www.cyveillance.com/us/newsroom/pressr/000710.asp>

<sup>4</sup><http://www.searchenginewatch.com/reports/sizes.html>

Informationen zur Vulkanfotografie auf Hawaii einholen, um Anlaufstellen für Einsätze auf dem Kilauea zu erhalten, während ein Vulkanologe sich über die seismische Aktivität Hawaiis der letzten 25 Jahre informieren möchte.<sup>5</sup>

Ohne Zusatzwissen über den Benutzer kann das System nicht wissen, ob ihn oberflächliche Informationen zur Plattentektonik im Nordpazifik, die Homepage des Vulkanfotografen-Clubs von Honolulu oder Untersuchungen von geographischen Instituten zur Seismologie Hawaiis mehr interessieren und das System die entsprechenden Seiten besser bewerten soll.

Durch intelligente Benutzeradaptivität kann der Suchraum deutlich eingeschränkt werden (im genannten Beispiel wären zum Beispiel Informationen zum beruflichen Hintergrund oder zu den Hobbies des Suchenden hilfreich). Aufgrund der Allgemeingültigkeit von Suchmaschinen ist es jedoch mit enormem Aufwand verbunden, diese Anforderungen tatsächlich umzusetzen und es bleibt die Frage, inwieweit in Zukunft andere Methoden wie die Spezialisierung von Suchmaschinen auf bestimmte (Fach-)Bereiche eher realisiert werden.

## 3.4 Adaptioneninformationen

Welche der Eigenschaften des Benutzers zur Adaption herangezogen werden, hängt in der Regel vom Einsatzgebiet ab, prinzipiell basiert aber die Anpassung auf fünf Hauptmerkmalen des Benutzers: Sein Wissen, seine Ziele, der Hintergrund, seine Erfahrungen sowie seine Präferenzen und Wünsche [5].

### 3.4.1 Wissen

Das Wissen des Benutzers ist bei weitem die wichtigste Informationsquelle für adaptive Hypermediasysteme; viele der in Abschnitt 3.6 vorgestellten Methoden basieren auf der Auswertung des aktuellen Wissensstands. Da sich das Wissen – insbesondere natürlich bei Lehrsystemen – ständig ändert, erfordert der Einsatz solcher Methoden eine dauernde Aktualisierung des internen Benutzermodells.

Meist wird ein Stereotypen- oder Overlaymodell verwendet, um das Wissen des individuellen Benutzers zu modellieren [13]; durch Kombination dieser beiden Methoden erhält man jedoch ein Modell, bei dem die Schwächen des einen Ansatzes durch den anderen kompensiert werden, vgl. Abschnitt 4.2.

### 3.4.2 Ziele

Die vom Benutzer verfolgten Ziele bei der Verwendung eines adaptiven Hypermediasystems hängen in erster Linie vom Einsatzgebiet ab; bei IR-Systemen ist das Ziel ein

---

<sup>5</sup>Zugegeben, das Beispiel ist etwas konstruiert, sollte aber die Problematik heutiger Suchmaschinen verdeutlichen.

Suchergebnis, bei Ausbildungssystemen hingegen ein Problemlösungs- oder Lernziel. Das verfolgte Ziel kann sich dabei schnell und oft ändern. Überwiegend sind es Methoden der adaptiven Navigationsunterstützung, die das Ziel des Benutzers analysieren (vgl. Abschnitt 3.6.3). Bei der Modellierung bietet sich die Unterscheidung zwischen High-Level- und Low-Level-Zielen an, beispielsweise bildet in einem Lehrsystem das Wissen eines Lektionsinhalts ein höheres Ziel als das Lösen einer Übungsaufgabe. In einigen Systemen wird eine Menge möglicher Ziele verwaltet, die das System anhand des Benutzerverhaltens wiedererkennen bzw. gewichten kann [5].

### 3.4.3 Hintergrund und Erfahrungen

Der Hintergrund umfasst alle Informationen zum Benutzer, die in irgendeiner Weise hilfreich zur Einschätzung des Benutzers sein können, angefangen von der Muttersprache über berufliche Hintergründe bis hin zur generellen Erfahrung im Umgang mit Computern (z. B. zur Anpassung der Benutzeroberfläche). Diese Informationen sind vom System nur schwer herleitbar und werden deshalb meist zu Beginn durch den Benutzer selbst angegeben. Erfahrungen des Benutzers beziehen sich in diesem Zusammenhang auf das Metawissen über die Domäne, also die Struktur und den Aufbau des vermittelten Inhalts. Implementiert werden Hintergrund und Benutzererfahrungen in der Regel durch Stereotypenmodelle; diese grobe Einteilung reicht in den meisten Fällen für diese Zwecke aus [5].

### 3.4.4 Präferenzen

Unter Präferenzen wird in diesem Zusammenhang die Adaptierbarkeit des Systems verstanden, d. h. die Möglichkeit für den einzelnen Benutzer, grundlegende Eigenschaften des Programms und der Benutzeroberfläche seinen Vorstellungen anzupassen. Die Benutzerpräferenzen spielen somit eine etwas andere Rolle als die bisher geschilderten Adaptioneninformationen.

Ein adaptives System kann versuchen, durch Verallgemeinerung und Zuordnung der Benutzereinstellungen zu Benutzer-Stereotypen sog. Benutzergruppenmodelle zu bilden [29]. Auf der Grundlage solcher Gruppenmodelle wird neuen Benutzern die Anpassung des Systems erleichtert und die Adaption besser abgestimmt. Auch für den Einsatz in kollaborativen Umgebungen sind diese Einstellungen wichtig; die Zusammenarbeit verschiedener Benutzer ist deutlich erschwert, wenn ihre Benutzermodelle zu sehr voneinander abweichen.

## 3.5 Adaptionsziele

Der Hauptzweck, der durch adaptive Systeme verfolgt wird, ist je nach Einsatzgebiet unterschiedlich; generell können jedoch die folgenden Ziele genannt werden, mit de-

nen Verbesserungen durch den Einsatz adaptiver gegenüber nicht-adaptiven Systemen erreicht werden können [47]:

**Effizienzsteigerung**

Beispielsweise die Minimierung des Suchaufwands bei adaptiven IR-Systemen, die Verringerung von Fehlerraten bei Lehrsystemen oder die Minimierung des Arbeitsaufwandes bei Informations- und Hilfesystemen.

**Effektivitätsverbesserung**

Bezieht sich auf die Verbesserung des Verständnisses und die vermehrte Interaktion in adaptiven Lehrsystemen.

**Akzeptanzverbesserung**

Verringerung der Ängstlichkeit und Steigerung der Motivation, vor allem in Informations- und Hilfesystemen, aber auch in Lehrsystemen.

## 3.6 Adaptionmethoden und -techniken

In adaptiven Hypermediasystemen können prinzipiell zwei Bereiche adaptiert werden: Der Inhalt der Seiten selbst („content-level adaptation“ oder *adaptive Präsentation*) und die Navigation zwischen den einzelnen Seiten („link-level adaptation“ oder *adaptive Navigationsunterstützung*) [5]. Brusilovsky unterscheidet dabei jeweils die Methoden und die Techniken; die Methoden beschreiben auf High-Level-Ebene, welche Möglichkeiten es gibt, während die Techniken auf einer implementierungsnäheren Ebene erklären, wie diese Methoden realisiert werden können. Diese Einteilung erscheint sinnvoll und wird in der Literatur häufig zur Beschreibung adaptiver Hypermediasysteme herangezogen, weshalb auch hier diese Trennung berücksichtigt wird.

Conlan führt außerdem die Bereiche „Structural Adaptation“ und „Historical Adaptation“ als Adaptionmethoden an [13]. Während ersterer weitgehend der globalen Orientierungshilfe von Brusilovsky (vgl. S. 21) entspricht, bietet letzterer eine zeitliche Orientierung (z. B. anhand des Lernfortschritts, des individuellen Lernpfades oder benutzerspezifischer Landmarken), die in der Klassifizierung nach Brusilovsky zwar nicht explizit in dieser Form genannt wird, jedoch letztendlich ebenfalls in den Bereich der globalen Orientierungshilfe fällt.

### 3.6.1 Adaptive Präsentation – Methoden

Adaptive Präsentation besteht bei den heutigen Systemen in erster Linie in der Anpassung des Textes an den individuellen Benutzer; die Adaption von Multimedia-Elementen wurde noch nicht in größerem Rahmen eingesetzt. Abbildung 3.1 zeigt den Zusammenhang zwischen den Methoden und Techniken zur adaptiven Textpräsentation.

	Conditional Text	Stretch-Text	Seitenvarianten	Frames
Zusätzl. Erklärungen	✓	✓		✓
Voraus./Vergleiche	✓	✓		✓
Erklärungsvarianten	✓		✓	✓
Fragment-Sortierung				✓

**Abbildung 3.1:** Die verschiedenen Methoden und Techniken zur Adaption der Textpräsentation. Die Haken zeigen an, mit welchen Techniken die einzelnen Methoden realisiert werden können (nach [5]).

### Zusätzliche Erklärungen

Dieser Methode liegt zugrunde, Teile der vorhandenen Information dem Benutzer vorzuenthalten, die für ihn momentan nicht von Interesse sind oder nicht zum besseren Verständnis des vermittelten Konzepts beitragen.

Beispielsweise können detaillierte Low-Level-Informationen für einen Anfänger irrelevant oder gar unverständlich sein, während es genau diese Informationen sind, die einen erfahrenen Benutzer interessieren; dieser wiederum benötigt weniger ausführliche Erklärungen zu grundlegenden Konzepten, die für unerfahrene Benutzer von entscheidender Wichtigkeit sind.

Generell bestehen die Erklärungen zu Konzepten also aus einem „Stammsatz“, der für alle Benutzer sichtbar ist, und mehreren optionalen Teilen, die nur für bestimmte Benutzergruppen sichtbar sind.

### Voraussetzungen und Vergleiche

Bei dieser Methode wird der Inhalt in Abhängigkeit vom Wissensstand des Benutzers unterschiedlich präsentiert: Je nach erforderlichen Vorkenntnissen werden vor der Erklärung eines Konzepts die Erklärungen aller Konzepte eingeblendet, die zum Verständnis erforderlich sind. Basis sind also die Vorbedingungen eines Konzepts. Grundlage kann jedoch auch die Ähnlichkeit von Konzepten sein; ist ein ähnliches Konzept dem Benutzer bereits bekannt, so erhält er eine entsprechende vergleichende Erklärung.

Diese Methode fördert vor allem die Einordnung neuer Begriffe in bereits vorhandenes Wissen.

### Erklärungsvarianten

Reicht das Ein- oder Ausblenden von Informationen nicht aus, so können mit dieser Methode verschiedene Varianten eines Dokuments oder Teilen davon ge-

speichert werden. Je nach Benutzermodell wird dazu die passende Variante vom System ausgewählt.

Auf diese Weise lässt sich beispielsweise ein Kurs in einem adaptiven Lehrsystem in verschiedenen Sprachen gestalten.

### **Sortierung**

Durch die Sortierung von Informationseinheiten kann sowohl das Wissen als auch der Hintergrund des Benutzers berücksichtigt werden. Wichtigere Teile werden weiter vorn platziert und erhalten so mehr Aufmerksamkeit vom Benutzer.

## **3.6.2 Adaptive Präsentation – Techniken**

### **Conditional Text**

Eine relativ einfach zu implementierende und dabei flexible Möglichkeit zur Anpassung des Inhalts ist, die zu präsentierenden Informationen in Teile zu zerlegen, deren Anzeige nur dann erfolgt, wenn alle Vorbedingungen erfüllt sind. Beispielsweise lassen sich so Teile mit redundanter oder unverständlicher Information ausblenden oder vergleichende Erklärungen ein- oder ausblenden.

### **Stretch Text**

Diese Methode dient ebenfalls dem Ein- und Ausblenden von Textteilen; hierbei wird jedoch irrelevante Information nicht komplett ausgeblendet, sondern auf einen Link „zusammengeschrunpft“, der bei Bedarf vom Benutzer wieder erweitert werden kann. Damit erhält der Benutzer die Möglichkeit, auch auf die Informationen zuzugreifen, die vom System als irrelevant eingestuft werden. Nachteil dieses Verfahrens ist die schwächere Führung des Benutzers im Vergleich zu Conditional Text.

### **Seiten- und Fragmentvarianten**

Die Methode der Erklärungsvarianten kann über Seiten- oder Fragmentvarianten implementiert werden. Bei Seitenvarianten werden die kompletten Informationseinheiten in verschiedenen Ausführungen gespeichert und vom System die passende ausgewählt, während bei den feiner gegliederten Fragmentvarianten ein Konzept aus verschiedenen Ausführungen von Teilkonzepten zusammengesetzt wird.

Mithilfe von Seitenvarianten lassen sich z. B. mehrsprachige Systeme auf einfache Weise realisieren.

### **Frame-basiert**

Der Frame- oder objektorientierte Ansatz ist der flexibelste aber auch aufwändigste. Jede Information zu einem Konzept wird als Objekt gespeichert, dessen Attribute mehrere Erklärungsvarianten, Verknüpfungen zu anderen Objekten, Beispiele, Übungen u. a. enthalten. Das System ermittelt zur Erklärung eines Konzepts die für den momentanen Wissensstand des Benutzers am besten geeigneten Attribute.

### 3.6.3 Adaptive Navigationsunterstützung – Methoden

Im folgenden werden die Möglichkeiten erläutert, mit denen die Navigation des Benutzers beeinflusst werden kann; die hier erwähnten Techniken der adaptiven Sortierung, Ausblendung und Annotation werden im nächsten Abschnitt beschrieben. Abbildung 3.2 zeigt die Umsetzung der einzelnen Methoden mithilfe der verschiedenen Techniken.

	Direkte Führung	Link-Sortierung	Link-Ausblendung	Link-Annotationen	Map-Adaption
Globale Führung	✓	✓			
Lokale Führung	✓	✓	✓	✓	✓
Lokale Orientierung		✓	✓	✓	✓
Globale Orientierung			✓	✓	✓
Personalisierte Sichten		✓	✓		

**Abbildung 3.2:** Die verschiedenen Methoden und Techniken zur adaptiven Navigationsunterstützung. Die Haken zeigen an, mit welchen Techniken die einzelnen Methoden realisiert werden können (nach [5]).

#### Globale Führung

Die globale Führung dient dem Folgen des kürzesten Pfads zu einem globalen Ziel; dies spielt vor allem in IR-, Informations- und Hilfesystemen eine wichtige Rolle, bei denen das Ziel vorwiegend im Finden einer gesuchten Information besteht. Realisiert werden kann globale Führung durch den direkten Vorschlag, welcher der auf einer Seite verfügbaren Links als nächstes verfolgt werden soll oder durch adaptives Sortieren der Verweise, was den Vorteil hat, dass auch die Relevanz anderer Links erkennbar ist.

Bei adaptiven Lehrsystemen ist das globale Ziel des Benutzers das Wissen über einen großen Teil des Inhalts; hier ist außerdem der sich ständig ändernde Wissensstand des Benutzers mit zu berücksichtigen. In der Regel wird ein dynamischer „Next“-Link eingesetzt, um den Kursinhalt zu linearisieren. Dazu wird häufig die Technik des „Curriculum sequencing“ aus dem Bereich der Intelligen-ten Tutorsysteme (ITS) eingesetzt, die dem Lerner einen individuell optimalen Lernpfad ermittelt [6].

#### Lokale Führung

Während die Grundlage der globalen Führung das Ziel des Benutzers ist, ist bei der lokalen Führung ein solches globales Ziel nicht erforderlich; vielmehr versucht das System, den Benutzer durch Analyse der Präferenzen, des Wissens und des

Hintergrunds zu führen. Beispiel ist die Sortierung der Links anhand von Benutzerpräferenzen in IR-Systemen.

### Lokale Orientierungshilfe

Das Ziel von lokaler Orientierungshilfe ist es, dem Benutzer ein Gefühl zu vermitteln, wie die nähere Umgebung der aktuell präsentierten Seite aussieht. In bestehenden Systemen werden dazu entweder irrelevante Links ausgeblendet oder zusätzliche Informationen zu vorhandenen Links präsentiert. Grundlage dafür sind Präferenzen, das aktuelle Ziel oder die Erfahrung des Benutzers – Anfänger erhalten somit wesentlich weniger Navigationsmöglichkeiten als erfahrene Benutzer, was der oftmals erkennbaren Hilflosigkeit von Neulingen aufgrund des ihnen fehlenden (Struktur-)Wissens über die Domäne entgegenkommt.

### Globale Orientierungshilfe

Mit einer globalen Orientierungshilfe wird dem einzelnen Benutzer die Vernetzung der Dokumente, also die Struktur der Domäne und der einzelnen Seiten, vermittelt. Durch Ausblenden von Links wird die Orientierung gefördert, da nur ein Teil der vorhandenen Dokumente betrachtet wird, der Benutzer also nicht mit dem vollen Umfang der Domäne, sondern nur mit einem Ausschnitt konfrontiert wird.

### Personalisierte Sichten

Eine neuere Forschungs- und Entwicklungsrichtung ist die Organisation eines adaptiven Systems für den Arbeitsalltag; in klassischen Systemen wird dies allein dem Benutzer überlassen, Webbrowser können z. B. mit Bookmarks personalisiert werden; weiter gehende Systeme ermöglichen eine Adaptierbarkeit auch anhand des Benutzermodells [49].

Adaptive Systeme, die also zusätzlich zur Adaptierbarkeit dem Benutzer bei der Personalisierung helfen, gibt es bisher nur wenige; ein einfaches Beispiel ist in MS Office 2000 realisiert: länger ungenutzte Menüeinträge werden auf einen einzelnen Menüpunkt zusammengeschrumpft und dieser kann bei Bedarf (vom Benutzer) wieder expandiert werden.

## 3.6.4 Adaptive Navigationsunterstützung – Techniken

Brusilovsky unterscheidet generell zwei Arten von Verweisen [5]: *kontextunabhängige Links* sind vom Inhalt der dargestellten Seite unabhängig (z. B. Verweise zum Inhaltsverzeichnis oder zur allgemeinen Bedienungshilfe), *kontextabhängige Links* tauchen im Inhalt der Seite auf (z. B. „Hot words“ im Text und „Hot spots“ in Grafiken). Während erstere leicht zu sortieren, auszublenden und mit Annotationen zu versehen sind, gilt für letztere, dass diese nicht sortiert werden können und dass das Ausblenden im Text häufig nicht erwünscht ist. Einige der folgenden Techniken sind dementsprechend nur auf eine der beiden Linkarten anwendbar.

### Direkte Führung

Die direkte Führung ist eine einfache Technik, das Netzwerk von Dokumenten im System individuell angepasst zu linearisieren; entweder indem der beste Link hervorgehoben wird oder durch einen zusätzlichen dynamischen Link zur nächsten Seite. Diese Technik ist leicht zu implementieren, jedoch wird nur relativ wenig Hilfestellung geleistet, wenn man – aus welchem Grund auch immer – dem Vorschlag des Systems nicht folgen möchte.

### Sortieren von Links

Die Idee von adaptiver Sortierung besteht darin, je nach Wissensstand des Benutzers die Verweise auf einer Seite nach ihrer Wichtigkeit zu sortieren; durch Platzierung weiter vorne in der Liste erhält ein Link zwangsläufig mehr Aufmerksamkeit. Der Einsatz ist allerdings beschränkt: Lediglich kontextunabhängige Links können sortiert werden. Ein weiteres Problem stellt die instabile Ordnung dar: Die Reihenfolge kann sich bei jedem Besuch der Seite ändern, was insbesondere bei Lehrsystemen und für Anfänger eher verwirrend als hilfreich ist. Haupteinsatzgebiete für adaptive Sortierung sind dementsprechend IR-, Informations- und Hilfesysteme, bei denen die Navigationsdauer durch eine Relevanzsortierung der Links erheblich verringert werden kann.

### Ausblenden von Links

Das Ausblenden von Links ist in adaptiven Hypermediasystemen sehr weit verbreitet und wird praktisch von jedem bestehenden System eingesetzt. Das Ziel ist offensichtlich: Die Verkleinerung des Navigationsraums durch Ausblenden von Links zu irrelevanten Seiten, womit die Gefahr des „Lost in Hyperspace“ reduziert wird. Außerdem ist das Erscheinungsbild stabiler als beim Sortieren von Links.

In adaptiven Lehrsystemen wird das Ausblenden von Links häufig bei den folgenden Verweiszielen eingesetzt: Verweise zu Lerneinheiten, die noch unerfüllte Vorbedingungen haben, also noch nicht gelernt werden sollten, und Verweise zu Seiten aus anderen Lektionen, die also nicht zum Erreichen des aktuellen Lernziels beitragen.

Das Ausblenden von Links kann auf verschiedene, einfache Weisen realisiert werden; De Bra unterscheidet in diesem Zusammenhang die folgenden Möglichkeiten [14]:

#### Link hiding

Der Linktext wird als normaler Text dargestellt, was eine spezielle Form der Link-Annotation darstellt (s. u.); sinnvoll nur in Zusammenhang mit kontextabhängigen Verweisen.

#### Link removal

Der Link wird vollständig ausgeblendet (beschränkt auf kontextunabhängige Verweise).

**Link disabling**

Der Link wird als solcher präsentiert, aber die eigentliche Verweisfunktion ausgeschaltet oder mithilfe von speziellen Link-Annotationen anders präsentiert. Dies ist mit beiden Arten von Links möglich.

**Link-Annotationen**

Mithilfe von Annotationen können Links zusätzliche Informationen über den Zustand des Zieldokuments erhalten; in Lehrsystemen beispielsweise, ob der Link zu einem lernbaren Konzept führt, oder ob noch nicht alle Voraussetzungen erfüllt sind.

Annotationsformen sind entweder kleine Piktogramme, unterschiedliche Farben, unterschiedliche Schriftgrößen oder -formatierungen. Der adaptive Lispkurs ELM-ART verwendet z. B. als Annotationen zum einen Punkte in den Ampelfarben vor den Links, um dem Lerner zu zeigen, welche Dokumente gelesen werden können und zum anderen verschiedene Schriftformatierungen, in erster Linie zur Unterstützung Farbenblinder [11], s. Abschnitt 4.5.1 für eine ausführlichere Beschreibung von ELM-ART.

Während mit adaptivem Ausblenden von Links nur die zwei Zustände „relevant“ und „nicht-relevant“ unterschieden werden können, können mithilfe von Annotationen auch feinere Abstufungen erfolgen. In Webbrowsern wird standardmäßig auf Basis der „Browsing history“ zwischen bereits besuchten und nicht besuchten Links unterschieden, eine hilfreiche Form der Navigationsunterstützung, die für adaptive Systeme aber nicht ausreicht.

Annotationen schränken die kognitive Überlastung des Benutzers nicht so sehr ein wie das rigorose Ausblenden irrelevanter Links, jedoch kann das Ausblenden durch „Dimmen“ der Links recht gut nachgeahmt werden, mit den Vorteilen, dass die Links bei Bedarf vom Benutzer verfolgt werden können und dass sich der Benutzer kein falsches mentales Modell der Domänenstruktur bildet.

**Site-Map Adaption**

Die bisherigen drei Techniken können auch zur Adaption von Site Maps verwendet werden, wodurch die eigentliche Struktur der Maps jedoch nicht geändert wird. Ansätze für eine Adaption der Struktur gibt es mehrere, die praktische Umsetzung ist allerdings nur vom System HYPERCASE bekannt [5].

## 3.7 Evaluationen adaptiver Techniken

Viele der im vorigen Abschnitt vorgestellten Techniken zur individuellen Anpassung an den Lerner wurden in bestehenden adaptiven Lehrsystemen implementiert und erfolgreich eingesetzt. Statistisch fundierte empirische Untersuchungen zur Akzeptanz und Hilfeleistung adaptiver Techniken sind jedoch nur wenige vorhanden; viele dieser Untersuchungen basieren vielmehr auf informellen Bewertungen der Benutzer.

Am besten untersucht wurde die Technik der Link-Annotationen in adaptiven Lehrsystemen. Brusilovsky und Pesin zeigten in einer Untersuchung des Systems ISIS-Tutor [10], dass die Zahl der benötigten Navigationsschritte und Wiederholungen bekannter Konzepte beim Einsatz adaptiver Annotationen signifikant geringer war als beim gleichen Inhalt ohne adaptive Techniken [18]. Gleichzeitig wurde in diesem Experiment kein signifikanter Unterschied zwischen Link-Annotationen und Ausblendung noch nicht lernbarer Konzepte gefunden.

In einer informellen Befragung zum System WEST-KBNS wurden 30 Universitätsstudenten zur Einschätzung der mit Annotationen versehenen globalen Site-Map aufgefordert [18]. Die ermittelte Durchschnittsnote betrug 3,9 von 5 bei einer Standardabweichung von 1,3. Diese allgemeine Einschätzung lieferte trotz fehlender statistischer Analyse ein positives Feedback für Link-Annotationen.

Bei einer Evaluation im Zusammenhang mit dem System ELM-ART II wurde der Einfluss von direkter Führung und Link-Annotationen auf die Motivation der Studenten untersucht [51]. Dabei wurde ein signifikanter Unterschied bei adaptiver direkter Führung gefunden, kein signifikanter jedoch bei Einsatz von Link-Annotationen.

Ferner wurde in demselben Experiment auch der Einfluss dieser beiden Techniken auf die Zahl der Navigationsschritte untersucht, wobei sich zeigte, dass keine eine signifikante Verringerung hervorbrachte, die direkte Führung aber einen im Vergleich zu den Link-Annotationen größeren Effekt hatte. Dieser Effekt ging jedoch im weiteren Verlauf des Experiments verloren; dies wurde mit der anfänglichen Unerfahrenheit der Testpersonen erklärt, die sich gerade zu Beginn auf adaptive Vorschläge des Systems verließen, bei andauernder Nutzung jedoch die Struktur der Domäne besser überblickten und nicht mehr auf die direkte Führung angewiesen waren.

Die Schlussfolgerungen dieser Untersuchung wurden aber bereits von Weber und Specht selbst als sehr vorsichtig eingestuft, da die Zahl der Testpersonen sehr gering war [51] und ferner äußere Faktoren wie Verbindungsgeschwindigkeit, allgemeine Computererfahrung u. a. nicht berücksichtigt wurden. Aus diesem Grund kann dieses Experiment nicht unbedingt gegen Link-Annotationen gewertet werden [18].

In einer neueren Untersuchung wurde der Einfluss von adaptiven Link-Annotationen auf das Lernen und die individuellen Lernpfade von Studenten mit dem System InterBook getestet [18], [7]. Dazu wurde die Nutzung des Systems von 25 Lehramtsstudenten über vier Wochen beobachtet, in denen den Testpersonen zwei Kapitel über ClarisWorks Datenbanken und Spreadsheets präsentiert wurden.

Nach einer allgemeinen Einführung mit anschließendem Test über die Eigenschaften von InterBook konnten die Testpersonen in der ersten Woche das System frei verwenden. Nach dieser ersten Phase wurden zwei zufällige Gruppen gebildet, von denen eine adaptive Link-Annotationen erhielt, die andere nicht. Anschließend hatten beide Gruppen eine Woche lang Zugriff auf das Kapitel über Datenbanken. In der dritten Phase absolvierten alle Testpersonen einen Multiple-Choice-Test über den Stoff des ersten Ka-

pitels, woraufhin sie wiederum für eine Woche Zugriff auf das Kapitel über Spreadsheets hatten; dabei wurde die Präsentation der Link-Annotationen der Gruppen gewechselt. In der letzten Phase wurde analog zum ersten Teil ein MC-Test über Spreadsheets durchgeführt.

Die Auswertung der Lernpfade und Testergebnisse lieferte schließlich keine signifikante Verbesserung durch Link-Annotationen, es wurde jedoch eine deutliche Korrelation zwischen dem Folgen von adaptiv vorgeschlagenen Lektionen und besseren Testergebnissen gefunden [7]. Weitere Ergebnisse dieser Studie waren, dass Anfänger durch Link-Annotationen häufiger eine nicht-sequentielle Reihenfolge, also einen anderen Link als den stets vorhandenen „Continue“-Link, wählten [18].

Interpretiert wurde das Ergebnis dahingehend, dass Link-Annotationen nur dann einen positiven Effekt haben, wenn sie als Hilfe akzeptiert werden und die empfohlenen Seiten tatsächlich angewählt werden – ansonsten überwiegt jedoch der negative Effekt kognitiver Strapazierung durch Einsatz weiterer informationstragender Einheiten.

In der Dissertation von Specht [47] wurde erneut der Einfluss von Link-Annotationen auf die Lerneffizienz untersucht. Verglichen wurden hier vier Benutzergruppen mit unterschiedlichen Annotationen kombiniert mit der „inkrementellen Einführung von Hyperlinks“: Gruppe 1 sah alle Annotationen, wobei Links, die als zu schwierig eingestuft wurden, jedoch nicht verfolgt werden konnten (Annotationen und inkrementell); Gruppe 2 sah keine Annotationen und nur empfohlene Links (nur inkrementell); Gruppe 3 sah alle Annotationen und konnte allen angebotenen Links folgen (nur Annotationen); Gruppe 4 sah keine Annotationen und konnte wie Gruppe 3 allen Links folgen (weder Annotationen noch inkrementell).

In einem Experiment mit 85 Versuchspersonen wurden die Anzahl der richtig beantworteten Fragen in einem Abschlusstest sowie die Lesedauer der präsentierten Inhalte dieser vier Gruppen miteinander verglichen. Es zeigte sich, dass hier Gruppe 1 bei beiden untersuchten Faktoren am besten abschnitt; das Ergebnis wurde so gedeutet, dass die Annotationen dem Benutzer zeigen, welche Wege prinzipiell möglich sind, er aber aufgrund des eingeschränkten Navigationsraums „auf dem richtigen Pfad“ bleibt.

Im nächsten Kapitel werden nach einer Beschreibung des generellen Aufbaus adaptiver hypermedialer Lehrsysteme exemplarisch einzelne dieser in der Praxis erfolgreich eingesetzten Systeme und die von ihnen eingesetzten adaptiven Techniken im Überblick vorgestellt. Den Abschluss bildet – als Vorgriff auf die Kapitel 6 bis 8 – eine entsprechende Einordnung des im Rahmen dieser Arbeit implementierten Systems iTeach.

# Kapitel 4

## Adaptive Lehrsysteme in der Praxis

Der generelle Aufbau von adaptiven Hypermediasystemen ist unabhängig vom Einsatzgebiet und Typ des Systems und besteht aus vier Komponenten. Durch die Interaktion des Benutzers mit einem adaptiven Hypermediasystem erhält das System Informationen über den Benutzer selbst, die im *Benutzermodell* gespeichert werden. Die *Inferenzkomponente* analysiert bei einer Aktualisierung des Benutzermodells, ob weitere Schlüsse über den Benutzer gezogen werden können und ändert seinerseits das Modell. Das Benutzermodell dient neben dem *Domänenmodell*, das die eigentlichen Informationen zum behandelten Themengebiet enthält, einer *Adaptionskomponente* als Grundlage für den eigentlichen Adaptionsvorgang.

Der gesamte Adaptionsprozess lässt sich also in die zwei Phasen *Benutzermodellierung* und *Adaption* unterteilen [5]. Abbildung 4.1 verdeutlicht diese Vorgänge und das Zusammenspiel der einzelnen Komponenten eines adaptiven Systems.

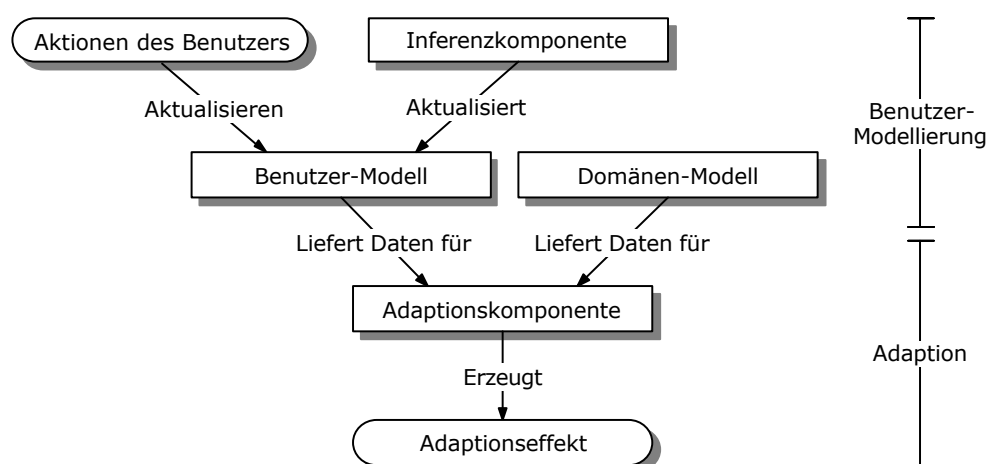


Abbildung 4.1: Die Komponenten eines adaptiven Hypermediasystems.

Im Folgenden werden diese einzelnen Komponenten näher erläutert; abschließend werden einige adaptive Lehrsysteme kurz vorgestellt, die aufgrund ihrer Praxistauglichkeit mit zum Design von iTeach beigetragen haben.

## 4.1 Domänenmodell

Nach De Bra lässt sich die in einem adaptiven Hypermediasystem behandelte Domäne auf drei Ebenen beschreiben, die sich in ihrem Abstraktionsgrad voneinander unterscheiden [14]:

1. Auf der niedrigsten Ebene befinden sich „atomare“ Einheiten, die die eigentlichen Informationen enthalten, z. B. ein Textabschnitt, eine Grafik, Tabelle etc. Diese *Fragmente* können entweder als statische Einheiten vorliegen oder dynamisch zur Laufzeit generiert werden (z. B. von einem Spracherzeugungsmodul).
2. Die mittlere Ebene besteht aus *Präsentationseinheiten*, in der Regel einzelne HTML-Seiten, die aus einzelnen Fragmenten zusammengesetzt sind. Die Adaptionskomponente ist dafür zuständig, die Fragmente auszuwählen, die in einer Webseite zusammen dargestellt werden (vgl. Abschnitt 4.3).
3. Die abstrakteste Ebene beschreibt eine Domäne als Menge von High-Level-Konzepten, die miteinander über verschiedene Verwandtschaftsbeziehungen in Zusammenhang stehen. Behandelt das System z. B. eine Programmiersprache, so kann das Konzept „Liste“ als (eine von mehreren) Ausprägungen des Konzepts „Abstrakte Datentypen“ aufgefasst werden.

Die Beschreibung auf dieser höchsten Ebene erfolgt häufig als sog. Konzeptnetz oder semantisches Netz [42]. Dabei werden die Konzepte als Knoten in einem Graphen aufgefasst, die Kanten im Graphen stellen die gegenseitigen Abhängigkeiten der Konzepte dar.

Für das Design des Domänenmodells ist es erforderlich, die behandelten Konzepte und die Relationen zwischen ihnen zu analysieren. Die meisten Systeme unterstützen dafür eine vordefinierte Menge an unterschiedlichen Relationen, wobei die gebräuchlichsten die folgenden sind:

- Vorbedingungen (*is-prerequisite-of*)
- Instanz-Beziehungen (*is-a*)
- Teilrelationen (*is-part-of*)
- Allgemeine Abhängigkeiten (*is-related-to*)

Die Adaption an den individuellen Benutzer wird von der Adaptionskomponente übernommen, die neben dem Wissen über die Domäne und deren Struktur natürlich auch Informationen über den Benutzer selbst benötigt.

## 4.2 Benutzermodell

Im Idealfall könnte ein adaptives Hypermediasystem komplett eigenständig Informationen über den Benutzer sammeln während dieser sich mit dem System beschäftigt. Dieser Fall ist im Bereich der adaptiven Hypermediasysteme jedoch unrealistisch [5], da letztendlich jede Interaktion zwischen Benutzer und System aus dem Folgen eines Verweises (und damit meistens die Anforderung von weiteren Informationen) besteht. Brusilovsky nennt in diesem Zusammenhang zwei Probleme bei der automatischen Benutzermodellierung [5]:

### Unzuverlässigkeit der Daten

Die Informationen, die dem System zugänglich sind, können zu falschen Schlüssen führen, z. B. bei der Aktualisierung des Wissensmodells und bei der eigentlichen Adaption.

### Ungenügende Informationen

Einige der benötigten Informationen wie Benutzerpräferenzen und Hintergrundwissen können nicht automatisch hergeleitet werden. Sollen diese Daten für den Adaptionsprozess herangezogen werden, ist die Angabe durch den Benutzer zwingend erforderlich.

Adaptive Hypermediasysteme können prinzipiell lediglich zwei Arten der Information verwenden, um Daten über den Benutzer zu erhalten: seine „Browsing history“ und das Zeitverhalten (wie lange eine bestimmte Seite im Browser angezeigt wurde).

HYPERFLEX, ein adaptives Kochbuch und Kombination von adaptivem Informations- und IR-System [29], benutzt beispielsweise diese Zeitdaten, um die Relevanz des betrachteten Knotens für das aktuell verfolgte Ziel zu beurteilen. Das Lehrsystem ISIS-Tutor versucht aus der Zahl der Besuche einer Seite zu schließen, wie bekannt dem Benutzer deren Inhalt ist [5]. Diese Daten sind für sich genommen jedoch äußerst unzuverlässig (z. B. kann der lange Aufenthalt auf einer Seite bedeuten, dass der Benutzer Probleme hat, das vermittelte Konzept zu begreifen oder dass gerade das Telefon klingelt . . . ).

In zwei Anwendungsbereichen adaptiver Hypermediasysteme können zusätzlich einigermaßen zuverlässige Quellen zur Informationsgewinnung herangezogen werden: In adaptiven Hilfesystemen kann anhand des Kontexts das Ziel des Benutzers hergeleitet werden, und in adaptiven Lehrsystemen geben die Ergebnisse von Übungsaufgaben und Tests deutlich zuverlässigere Auskunft über den tatsächlichen Wissensstand eines Benutzers. In anderen Anwendungsbereichen muss sich das System für weiter gehende Informationen aber auf explizite Angaben des Benutzers selbst verlassen.

Die eigentlichen Techniken zur Modellierung eines einzelnen Benutzers in adaptiven Systemen entstammen dem Gebiet der intelligenten Tutorsysteme (ITS); durchgesetzt haben sich vor allem die Ansätze der Modellierung durch Stereotypen und mittels Overlay-Modellen bzw. Kombinationen dieser beiden Ansätze [13].

### 4.2.1 Stereotypen

Die Einteilung von Benutzern in Benutzergruppen, sog. Stereotypen, hat sich in der Praxis besonders in solchen Anwendungsfällen als guter Ansatz erwiesen, in denen eine schnelle, aber nicht notwendigerweise vollständig korrekte Einteilung erforderlich ist [39], [30].

Kobsa nennt drei aufeinander aufbauende Anforderungen für eine angemessene Modellierung:

#### **User subgroup identification**

Es müssen die verschiedenen Benutzergruppen identifiziert werden.

#### **Identification of key characteristics**

Für die einzelnen Stereotypen müssen die charakteristischen Eigenschaften analysiert und identifiziert werden.

#### **Representation in (hierarchically ordered) stereotypes**

Die verschiedenen Benutzergruppen müssen in einem passenden System formal repräsentiert werden. Dabei bieten sich Hierarchien an, um Ähnlichkeiten und Spezialisierungen verschiedener Gruppen effizienter modellieren zu können.

Beim Einsatz von Stereotypen zur Benutzermodellierung wird häufig eine Einteilung der Benutzer anhand ihres Wissens über ein bestimmtes Themengebiet vorgenommen, beispielsweise in Anfänger, Fortgeschrittene und Experten [30]. Der Nachteil von Benutzermodellierung anhand von Stereotypen liegt auf der Hand: Es besteht vor allem die Gefahr einer zu groben Einteilung und dementsprechend eine ungenaue Adaptivität. Dies ist der Grund, weshalb einige adaptive Systeme stattdessen genauere Overlaymodelle zur Benutzermodellierung einsetzen.

### 4.2.2 Overlay-Modell

Bei der Benutzermodellierung anhand eines Overlay-Modells wird das Wissen eines Benutzers als Teilmenge des (idealen) Wissens eines Experten repräsentiert [44]. Dazu wird im Benutzermodell für jedes Konzept der Wissensstand über dieses Konzept gespeichert (im einfachsten Fall ist dies ein simples Abhaken der bekannten Konzepte).

Der größte Nachteil von Overlay-Modellen ist die Tatsache, dass das Modell nicht unterscheiden kann, ob dem Lerner Wissen fehlt oder ob er andere Lernstrategien verfolgt, um ein Lernziel zu erreichen [44]. Der Nachteil ist im Bereich der intelligenten Tutorssysteme von größerer Wichtigkeit als in adaptiven Hypermediasystemen, da bei letzteren explizite Lernstrategien aufgrund der schlechten Beobachtbarkeit des Benutzers nur begrenzt analysiert werden können.

In einigen adaptiven Lehrsystemen erfolgt eine Einstufung je nachdem, ob eine Seite nur gelesen wurde oder auch enthaltene Übungen und Tests absolviert wurden (z. B. in die Kategorien `READ`, `KNOWN` und `WELL_KNOWN`). Das System von Pilar da Silva et al. erlaubt eine sehr viel feinkörnigere Beurteilung [38], vgl. auch Abschnitt 4.5.4.

Häufig werden diese beiden vorgestellten Modellierungsarten in adaptiven Hypermediasystemen zu einem hybriden Modell kombiniert [13]: Die Initialisierung des Benutzermodells erfolgt durch Zuordnung in ein Stereotypenschema und im weiteren Verlauf der Nutzung wird mithilfe eines Overlaymodells diese grobe Klassifizierung verfeinert. Vorteil dieses Ansatzes ist es, dass das System mit einem besseren „Default“ beginnt, so dass bereits zu Beginn eine bessere Adaptivität erzielt werden kann.

### 4.3 Adaptionskomponente

Die Adaptionskomponente dient der eigentlichen Anpassung der präsentierten Informationen an den einzelnen Benutzer. Sie greift auf die Modelle der Domäne und des Benutzers zurück, um Inhalt und Struktur des Inhalts anzupassen (vgl. Abbildung 4.1). Dazu setzt sie die in Abschnitt 3.6 vorgestellten Techniken zur Adaption der Präsentation und zur Navigationsunterstützung um.

### 4.4 Inferenzkomponente

Die Inferenzkomponente versucht aus den Informationen über den Benutzer – seien diese entweder explizit vom Benutzer selbst angegeben oder implizit aus den Interaktionen zwischen Benutzer und System hergeleitet – zusätzliche Informationen herzuleiten, die aus den erhaltenen Daten nicht direkt ersichtlich sind. Die Inferenzkomponente kann also als Menge logischer Regeln aufgefasst werden, die im Anschluss an eine Aktualisierung des Benutzermodells ausgeführt werden.

Bei einem Großteil der adaptiven Lehrsysteme werden diese Regeln ausgeführt, wenn das System Informationen über den geänderten Wissensstand des Benutzers erhält, vgl. z. B. [51], [31], [38]. Dazu wird eine Vorwärtsinferenz gestartet, die überprüft, ob mit diesem neuen Wissen neue bzw. andere Konzepte gelernt werden können, während eine Rückwärtsinferenz in analoger Weise eventuell vorhandene Vorbedingungen überprüft und ggf. als bekannt einstuft; bei Wissensabnahme (also einem nicht-monotonen Wissensmodell, das auch das Vergessen von Konzepten und Zusammenhängen berücksichtigt) wird eine entsprechende Rücknahme von Wissen vorgenommen [15].

Das Lehrsystem KBS Hyperbook berücksichtigt Unsicherheiten im Wissen des Benutzers, indem es das Wissen des individuellen Lernalers in Form bedingter Wahrscheinlichkeiten (sog. „knowledge vectors“) modelliert [25]; die gesamte Domäne wird dabei als Bayessches Netz [42] modelliert. Die Inferenzkomponente ermittelt auf dieser Basis Wahrscheinlichkeitsverteilungen für jedes Konzept der gesamten Domäne (vgl. auch Abschnitt 4.5.3).

Der Vorteil dieses Ansatzes liegt in der Sache selbst, nämlich die Möglichkeit, Unsicherheiten bei der Beobachtung des Studenten und der Analyse seines Verhaltens zu berücksichtigen. Ferner ermöglicht dieser Ansatz eine (nahezu) beliebig feine Granularität bei der Modellierung des Wissens eines Lernalers [25]. Diese Vorzüge werden jedoch

mit einem komplexeren Domänenmodell erkaufte – es sind die bedingten Wahrscheinlichkeiten für die einzelnen Konzepte erforderlich, die in der Regel von Experten ermittelt werden.

## 4.5 Vergleich bestehender Lehrsysteme

Ähnlich der in Kapitel 2 geschilderten zunehmenden Nutzung des WWW als Informationsplattform bei nicht-adaptiven Lehrsystemen lässt sich auch bei adaptiven Lehrsystemen eine solche Tendenz erkennen, die die Fähigkeiten von WBT- und AH-Systemen kombiniert.

Um den Stand der Entwicklung aufzuzeigen, werden in den folgenden Abschnitten einige im Internet verfügbare adaptive Lehrsysteme anhand ihrer adaptiven Eigenschaften vorgestellt. Diese Systeme hatten entweder direkt oder indirekt Einfluss auf die Entwicklung des im Rahmen dieser Arbeit implementierte System iTeach. Dementsprechend wird auch iTeach als Vorgriff auf Kapitel 6 kurz vorgestellt; den Abschluss bildet eine Übersicht über die in den verschiedenen Systemen implementierten adaptiven Techniken.

### 4.5.1 ELM-ART II

Das an der Universität Trier entwickelte System ELM-ART (Episodic Learner Model – Adaptive Remote Tutor) und seine Nachfolger ELM-ART II und InterBook basieren auf dem Stand-alone System ELM-PE, einem Einführungskurs in die Programmierung mit LISP [51].

Das System benutzt ein Episodic Learner Model, um den Lerner zu modellieren. Dazu wird das Wissen über den Benutzer in Form von einzelnen Episoden gesammelt, die vergleichbar sind zu Fällen beim fallbasierten Schließen [25]. Zur Konstruktion des Modells wird der vom Lerner in einer Übung erzeugte Programmcode anhand des Domänenmodells und einer Aufgabenbeschreibung analysiert. Eine genauere Beschreibung dieser Problemlösungskomponente enthält [51].

Das Domänenmodell besteht aus einem hierarchischen Netzwerk von Kurseinheiten, Lektionen, Konzepten und HTML-Seiten. Jede dieser Einheiten enthält einen Slot für den eigentlichen Seiteninhalt und eine Liste von Verweisen zu in Beziehung stehenden anderen Konzepten. Die HTML-Seiten enthalten zusätzliche Test-Slots, Seiten mit Problemstellungen (in der Regel Programmieraufgaben) und die für die erwähnte ELM-Analyse benötigte Beschreibung des Problems. Wurde eine Seite vom Benutzer angefordert, wird das entsprechende Konzept im Benutzermodell als bekannt markiert.

ELM-ART und seine Nachfolgesysteme stellen als adaptive Techniken die direkte Führung des Benutzers und Link-Annotationen zur Verfügung. Die Annotationen bestehen aus farbigen Punkten vor jedem Link, dessen Farbe in Analogie zu einer Ampel Auskunft über den Zustand des Verweisziels gibt; grün steht dabei für empfohlene, gelb in der Regel für neutrale (d. h. bereits gelesene) Seiten, während ein roter Punkt

bedeutet, dass dem Benutzer (noch) Wissen fehlt, um den Inhalt dieser Seite verstehen zu können. Er kann dem Link aber dennoch folgen und im Falle der korrekten Lösung von Tests bzw. Programmieraufgaben erfolgt eine Rückwärtsinferenz, die alle erforderlichen Konzepte als bekannt einstuft.

### 4.5.2 InterBook

InterBook stellt eine Weiterentwicklung von ELM-ART dar, bei der die Integration von Autorensystem und adaptivem Lehrsystem im Vordergrund stand [8]. Grundlage sind sog. *electronic textbooks*, die letztlich beliebig hierarchisch gegliederte Hypermediadaten darstellen. Das Domänenmodell besteht wie bei ELM-ART aus einem Konzeptnetz, wobei jedem Inhaltsabschnitt ein oder mehrere Konzepte gemäß ihrer „Rolle“ zugeordnet sind; eine Rolle ist dabei entweder Ergebnis (das Konzept wird in diesem Abschnitt erklärt) oder Vorbedingung (das Konzept wird für das Verständnis dieses Abschnitts benötigt).

Neben der Repräsentation des Wissens eines individuellen Lernalters in einem klassischen Overlaymodell wird zusätzlich eine Folge von Konzepten als Lernzielfolge gespeichert, die der Reihe nach vorgeschlagen werden.

Das Glossar stellt eine Visualisierung des Domänenmodells dar und kann als globale Site-Map aufgefasst werden; die Knoten in diesem Netzwerk stellen die einzelnen Konzepte, die Kanten zwischen den Konzepten Navigationspfade dar. Die Konzeptknoten enthalten Links zu allen Buch-Kapiteln und -Abschnitten, die das jeweilige Konzept erklären oder benötigen [8].

InterBook verwendet neben allgemeinen Navigationshilfen individuelle direkte Führung und wie ELM-ART adaptive Link-Annotationen nach der Ampel-Semantik, bei der bereits besuchte Seiten zusätzlich abgehakt werden. Ferner verfügt das System analog zu ELM-ART über sog. „prerequisite-based help“. Über einen entsprechenden Link erhält der Lerner Links zu allen erforderlichen Vorbedingungen der aktuellen Seite, so dass er bei Bedarf zu diesen Seiten springen und sein Wissen auffrischen kann.

### 4.5.3 KBS Hyperbook

KBS Hyperbook ist ein Werkzeug zur Organisation und Benutzung adaptiver, offener Hypermediasysteme im WWW von Henze und Nejd [25]. Offen bedeutet in diesem Zusammenhang, dass verteilte Informationen im Internet in das System bzw. die Kurse integriert werden können. Neben dem Wissen des Lernalters werden ebenso wie in InterBook die Lernziele des Benutzers für die individuelle Adaption berücksichtigt. Als Beispielinhalt wurde ein Einführungskurs in die Programmierung mit Java entwickelt, dessen Grundlage das Java Tutorial von Sun ist.

Das Domänenmodell besteht aus einem Konzeptnetz, das die Konzepte und ihre Abhängigkeiten in Bezug auf die Lernreihenfolge als sog. „knowledge items“ repräsentiert. Die einzelnen Informationseinheiten des eigentlichen Inhalts basieren auf semantischen

tischen Zusammenhängen (nicht syntaktischen wie Kapitel oder Abschnitt), die zueinander in Beziehung stehen; über diese Beziehungen wird die Linkstruktur der Domäne erzeugt. Jedes knowledge item wird durch mindestens eine Informationseinheit erklärt [26].

Die gesamte Nutzung des Systems basiert auf Projekten, die z. B. Programmier-Aufgaben oder Beispiele gelöster Aufgaben enthalten. Die Projekte sind ebenso wie die Informationseinheiten durch knowledge items indiziert, wodurch das System automatisch adaptiv annotierte Links zu relevanten Informationseinheiten anbieten kann (verwendet wird dazu wie bei ELM-ART und InterBook die Ampel-Metapher). Diese Links werden adaptiv sortiert, wodurch eine direkte Führung des Lernalters entsteht.

Der Benutzer kann ferner selbst mehrere knowledge items in Form eines Lernziels angeben, anschließend ermittelt das System eine Folge von Projekten, sodass dieses Lernziel erreicht wird; auch dies ist letztendlich eine direkte Führung, die dementsprechend auf verschiedenen Ebenen umgesetzt wird [26].

Das Studentenmodell ist ein Overlaymodell, wobei das Wissen über ein knowledge item mittels vier Stufen beschrieben wird: Experte, Fortgeschrittener, Anfänger, Neuling. Durch Verwendung von Wahrscheinlichkeitsverteilungen anstatt diskreter Werte ist eine feine Abstufung möglich. Implementiert ist das Studentenmodell als Bayessches Netz [42]. Die Knoten sind dabei die einzelnen knowledge items; Abhängigkeiten zwischen den knowledge items sind bedingte Wahrscheinlichkeiten [26].

Die Beobachtung des Benutzers ist auf die Projekte beschränkt, weder besuchte Seiten noch die „Browsing history“ werden analysiert. Feedback erhält das System entweder durch direkte Nachfrage und Beurteilung des Lernalters selbst (abgestuft von „einfach“ bis „zu schwer“) oder durch Analyse der Projektergebnisse von einem Experten.

#### 4.5.4 AHM

Das System AHM von Pilar da Silva et al. [38] ermöglicht im Gegensatz zu den bisher betrachteten Systemen eine wesentlich genauere Wissensmodellierung. Anstatt im Benutzermodell lediglich zwischen einzelnen abstrakten High-Level-Zuständen wie UNKNOWN, KNOWN und LEARNED und WELL\_LEARNED zu unterscheiden (vgl. z. B. ELM-ART), können Werte von 0 bis 99 angenommen werden [37]. AHM war das erste System, das eine solche Wissensmodellierung ermöglichte.

Die entsprechend fein granulare Domänen-Modellierung basiert auf einem Konzeptnetz mit gewichteten Vorbedingungs-Kanten; die Gewichtung ist ein Schwellwert, der überschritten werden muss, um dieser Kante folgen zu können [38]. Ein Konzept wird in der Regel von mehreren Dokumenten (Webseiten) zu einem gewissen Prozentsatz erklärt, der als eine Art Schwierigkeitsgrad aufgefasst werden kann. Ein Lerner kann dementsprechend ein Konzept nur lernen, wenn er alle dessen Vorbedingungen genügend kennt, also so viele Dokumente für ein Konzept gelesen hat, dass der jeweilige Schwellwert überschritten wurde.

Als adaptive Technik wird das Ausblenden von Links eingesetzt. Das System hat jedoch

noch Prototyp-Charakter, was u. a. daran zu erkennen ist, dass lediglich der Kanten-typ „ist-Vorbedingung-von“ unterstützt wird und die Dokumente jeweils nur ein Konzept erklären (können). Dennoch ist der Ansatz eines feiner granulären Wissensmodells äußerst viel versprechend – der Nachteil ist jedoch das erforderliche Expertenwissen, um die Kanten-Gewichtungen und Schwierigkeitsgrade der Dokumente anzugeben. De Bra nennt ferner folgende Punkte, die bei einer Umsetzung beachtet werden müssen [14]:

- Wird eine Seite mehr als einmal gelesen, sollte sie jedoch nur einmal zur Kenntnis des Konzepts beitragen.
- Wird eine Seite gelesen, deren Konzept-Vorbedingungen noch nicht vollständig erfüllt sind, sollte der Beitrag dieser Seite geringer sein, als wenn alle Vorbedingungen erfüllt sind. Eine entsprechende Verrechnung kann sich als schwierig erweisen, wenn der Benutzer die Seite zum ersten Mal bei nicht-erfüllten Vorbedingungen liest und später, wenn sie erfüllt sind.
- Zwei Seiten können die gleichen Informationen enthalten, sodass eine einfache Addition der Seitenbeiträge unzureichend ist.
- Die Summe der Seitenbeiträge kann 100% übersteigen, das Wissen über ein Konzept jedoch nicht.

Das im Rahmen dieser Arbeit implementierte System iTeach berücksichtigt wie AHM eine feinere Wissensmodellierung. Wie die genannten Schwierigkeiten umgangen wurden bzw. entkräftet werden können, wird in Kapitel 8 beschrieben.

#### 4.5.5 AHA

Das von De Bra und Calvi entwickelte System AHA („Adaptive Hypermedia Architecture“) ist ein auf XML und Java Servlets basierendes adaptives Lehrsystem mit verschiedenen Kursinhalten [16], [15]. Das Domänenmodell besteht aus Fragmenten, Seiten und abstrakten Konzepten; jede Seite ist dabei eine XML-Datei bestehend aus Fragmenten, die ein- oder ausgeblendet werden können. AHA unterscheidet neben normalen (HTML-)Links die Konzept-Relationen Ergebnis und Vorbedingung. Ähnlich wie AHM besteht das Benutzermodell aus einem Overlaymodell, das Werte von 0 bis 100 für einzelne Konzepte akzeptiert.

Die Vorbedingungen auf Konzeptebene stehen in einer separaten XML-Datei, und haben z. B. die folgende Form [15]:

```
<concept>
  <conceptname>c1</conceptname>
  <relationexpression>req1>30 and req2<80</relationexpression>
</concept>
```

Außer `and` werden auch die logischen Verknüpfungen `or` und `not` unterstützt. Die Regeln zur Steuerung der Anzeige von einzelnen Fragmenten sehen ähnlich aus und werden direkt im Inhaltsfile angegeben. Wird eine Seite angezeigt, wird das Benutzermodell anhand von „generate rules“ (diese stehen ebenfalls in einem separaten XML-File) aktualisiert. Eine solche Update-Regel sieht z. B. wie folgt aus:

```
<genitem>
  <name>c1</name>
  <genlist>c2:+40 c3:-30 c4:50</genlist>
</genitem>
```

Dies bedeutet, dass bei einer Zunahme des Wissens von Konzept `c1` um  $x\%$  das Wissen von Konzept `c2` um 40% von  $x$  erhöht, der Wert von `c3` um 30% von  $x$  erniedrigt und der Wert von `c4` auf 50 gesetzt wird. Es werden also sowohl relative als auch absolute Änderungen ermöglicht.

Neben der adaptiven Präsentation durch Conditional Text verwendet AHA Link-Annotationen (durch Änderung der Farbe), um die drei Linkarten „empfohlen“, „neutral“ und „nicht-empfohlen“ zu unterscheiden. Die Farben sind so gewählt, dass nicht-empfohlene Links wie normaler Text in schwarz dargestellt werden, wodurch eine faktische Ausblendung der Links stattfindet.

### 4.5.6 iTeach

Das System iTeach, das im Rahmen dieser Arbeit entwickelt wurde, ist ein System, mit dem es möglich ist, Kurse im WWW adaptiv zur Verfügung zu stellen. Grundlegendes Datenformat ist XML; als Frameworks werden Cocoon und Avalon der Apache Software Foundation<sup>1</sup> eingesetzt (vgl. Kap. 8 ab Seite 71).

Das Domänenmodell ist als Konzeptnetz realisiert, wobei die Struktur in einer XML-Datei angegeben wird. Diese Daten umfassen neben den Konzepten (Knoten) und Konzept-Beziehungen (Kanten) auch die Beziehungstypen, sowie die Schranken für die Wissensmodell-Granularität. Damit ist es noch allgemeiner gehalten als die vorgestellten Systeme, da je nach Größe und Komplexität der Domäne die Wissensmodellierung feiner oder gröber erfolgen kann.

Der Kursinhalt besteht aus einzelnen Fragmenten, von denen für die Erklärung eines bestimmten Konzepts mehrere ausgewählt und zusammen auf einer Webseite angezeigt werden. Die Umsetzung nach HTML erfolgt als Teil des Cocoon-Frameworks mit XSL(T), sodass das Layout streng vom Inhalt und der Programmlogik getrennt ist.

Das Studentenmodell ist ein klassisches Overlay-Modell, das zu Beginn optional über eine zusätzliche Stereotypen-Zuordnung anhand einer Benutzerbefragung initialisiert werden kann. Auf diese Weise kann besser auf Benutzer mit unterschiedlichem Hintergrundwissen eingegangen werden.

---

<sup>1</sup><http://www.apache.org>

Die Regeln für die Aktualisierung des Studentenmodells ähneln denen von AHA, jedoch ist die XML-Struktur flexibler, da die Zeichenfolgen nicht noch zusätzlich geparkt werden müssen.

Die adaptiven Techniken umfassen Conditional Text, Stretch Text, individuelle direkte Führung sowie Link-Annotationen.

Eine ausführliche Beschreibung des Systems folgt in den Kapiteln 6 bis 8.

#### 4.5.7 Übersicht über die verwendeten adaptiven Techniken

Tabelle 4.1 enthält als Zusammenfassung eine Übersicht der in den verschiedenen Systemen implementierten adaptiven Techniken. Als Vorgriff auf Kapitel 6 enthält die letzte Zeile die Techniken, die in dem in dieser Arbeit entwickelten System eingesetzt werden.

**Tabelle 4.1:** Eingesetzte Techniken in bestehenden adaptiven Lehrsystemen.

System	Content Adaptation	Navigation Support
ELM-ART II	–	Dir. Führung, Link-Annot.
InterBook	–	Dir. Führung, Link-Annot.
AHA	Cond. Text	Link-Annot.
KBS Hyperbook	–	Dir. Führung, Link-Annot.
AHM	–	Link Hiding
iTeach	Cond. Text, Stretch Text	Dir. Führung, Link-Annot.

# Kapitel 5

## Beispielinhalte

Während der frühen Entwicklungsphase von iTeach wurde ein Kriterienkatalog erstellt, um bereits im Internet verfügbares Kursmaterial auf seine Einsatzfähigkeit in einem adaptiven Lehrsystem untersuchen zu können. Im Vordergrund stand dabei die Untersuchung von frei erhältlichen Kursen zur Programmierung in Java, da dies aus verschiedenen Gründen einen reizvollen Inhalt darstellen würde:

- Die Programmiersprache Java hat seit ihrer ersten Veröffentlichung im Jahr 1996 eine stürmische Entwicklung erfahren – im universitären Umfeld erfreut sie sich aufgrund ihrer konsistenten Strukturierung und dem objektorientierten Aufbau als gute „Lernsprache“ und im Bereich der Internet-Programmierung verdrängen Java Servlets zunehmend herkömmliche CGI-Scripts.
- Außer dem adaptiven Lehrsystem KBS Hyperbook der Universität Hannover ist kein adaptiver Kurs über Java bekannt – im Gegensatz dazu gibt es eine große Anzahl nicht-adaptiver Tutorials.
- Es ist geplant, iTeach in Kombination mit einem Java Online-Praktikum im Rahmen der Virtuellen Hochschule Bayern einzusetzen. Dieses System wird zur Zeit in Würzburg am Lehrstuhl für Informatik II entwickelt [20].

Nach der Evaluation anhand des im folgenden beschriebenen Kriterienkatalogs wurde jedoch der Einsatz von iTeach dahingehend konkretisiert, dass als Inhalt für das Java Online-Praktikum eigenes Material zur Verfügung gestellt wird. Aufgrund der allgemein gehaltenen Framework-Architektur von iTeach ist jedoch der Einsatz nicht auf Java beschränkt; für die Evaluation von anderen potenziellen Kursinhalten ist der Katalog gut geeignet, da er alle Bereiche für die Beurteilung von Hypermedia-Kursen abdeckt.

### 5.1 Der Kriterienkatalog

Für die Evaluation von frei zugänglichen Java-Tutorials im Internet wurden etwa 15 solcher Kurse betrachtet; das Spektrum reichte von Vorlesungsaufzeichnungen über Pro-

grammierkurse mit Java bis hin zu kompletten Büchern.

Nach einer ersten Vorsortierung wurden schließlich zehn in Frage kommende Kurse genauer untersucht. Tabelle 5.1 enthält eine Aufstellung der evaluierten Kurse, anschließend werden die einzelnen Kriterien vorgestellt. Die ausführlichen Beurteilungen der einzelnen Kurse sind im Anhang A ab Seite 90 aufgeführt.

Der Kriterienkatalog umfasst insgesamt 41 Kriterien aus den Bereichen *Allgemeines*, *Kursinhalt*, *Präsentation* und *Bedienung*. Als Grundlage dienten zwei bereits verfügbare Kataloge ([45], [41]), die auf den hier erforderlichen Kontext angepasst wurden. Dabei wurde Wert darauf gelegt, die Kriterien möglichst allgemein zu halten, sodass auch die Beurteilung von Online-Kursen anderer Fachgebiete mit dieser Aufstellung möglich ist.

**Tabelle 5.1:** Evaluierte Javakurse. Die Nummern dienen der Identifizierung im Anhang.

Nr.	Autoren	Titel und URL
1	Diverse (Sun)	The Java Tutorial <a href="http://java.sun.com/docs/books/tutorial/index.html">http://java.sun.com/docs/books/tutorial/index.html</a>
2	D.J. Eck	Introduction to Programming using Java <a href="http://math.hws.edu/javanotes/">http://math.hws.edu/javanotes/</a>
3	B. Eckel	Thinking in Java 2 (Revision 10) <a href="http://www.mindview.net/Books/TIJ/">http://www.mindview.net/Books/TIJ/</a>
4	J. Gosling; H. McGilton	The Java Language Environment – A White Paper <a href="http://java.sun.com/docs/white/langenv/index.html">http://java.sun.com/docs/white/langenv/index.html</a>
5	B. Kjell	Introduction to Computer Science using Java <a href="http://chortle.ccsu.ctstateu.edu/cs151/cs151java.html">http://chortle.ccsu.ctstateu.edu/cs151/cs151java.html</a>
6	G. Krüger	Goto Java2 <a href="http://www.javabuch.de/">http://www.javabuch.de/</a>
7	L. Lemay	Java in 21 Tagen <a href="http://www.mut.de/media/buecher/Java2/index.htm">http://www.mut.de/media/buecher/Java2/index.htm</a>
8	V. Mukhi et al.	Shlurrrpp . . . . . Java <a href="http://edessa.topo.auth.gr/~thalis/java0.html">http://edessa.topo.auth.gr/~thalis/java0.html</a>
9	M. Pawlan	Essentials of the Java Programming Language 1 + 2 <a href="http://java.sun.com/developer/onlineTraining/Programming/">http://java.sun.com/developer/onlineTraining/Programming/</a>
10	D. Schmidt	Programming Principles in Java <a href="http://www.cis.ksu.edu/~schmidt/CIS200/">http://www.cis.ksu.edu/~schmidt/CIS200/</a>

### 5.1.1 Allgemeine Kriterien

Die allgemeinen Kriterien beziehen sich auf den Kurs selbst und die angesprochene Zielgruppe. Es wurden die folgenden vier Kriterien betrachtet:

**Erforderliche Vorkenntnisse**

Über welches Wissen sollte der Leser bereits verfügen? An welche Zielgruppe richtet sich der Kurs?

**Der Zielgruppe angemessene Präsentation**

Werden die Inhalte angemessen dargestellt? Beispielsweise sollten für Anfänger möglichst viele einfache Beispiele angeführt werden und mithilfe von Grafiken und Flussdiagrammen komplexere Vorgänge veranschaulicht werden.

**Umfang des Kurses**

Wie viele Themen umfasst der Kurs insgesamt? Wird das Domänenwissen anhand der Themenzahl erschöpfend behandelt?

**Sprache**

In welcher Sprache ist der Kurs gehalten?

**5.1.2 Kriterien zum Kursinhalt**

Anhand der folgenden elf Kriterien kann die Vermittlung des Wissens durch den Kurs beurteilt werden.

**Vermittlung von praktischem und theoretischem Wissen**

Erfährt der Benutzer sowohl Hintergründe als auch direkt in die Praxis umsetzbares Wissen?

**Die Inhalte werden korrekt und umfassend vermittelt**

Werden die einzelnen Themen ausführlich behandelt?

**Die beschriebene JDK-Version und Aktualität sind klar erkennbar**

Ist im Vorwort oder in der Einleitung das Datum der letzten Aktualisierung sowie die beschriebene JDK-Version angegeben?

**Die Texte sind knapp und präzise formuliert**

Sind die Texte nicht zu ausschweifend und langatmig geschrieben?

**Es werden (Code-)Beispiele zur Verdeutlichung eingesetzt**

Enthält der Kurs Beispiele von Java-Code und ist dieser syntaktisch korrekt?

**Die verwendeten Beispiele sind aussagekräftig**

Verdeutlichen die verwendeten Beispiele den Text und die erklärten Konzepte?

**Am Ende einer Lektion werden Übungen zur Vertiefung angeboten**

Werden am Ende einer Lektion Übungen angeboten? Welche Arten werden angeboten (Fragen zum Verständnis, Programmieraufgaben, Quiz)?

**Die Übungen sind motivierend und abwechslungsreich**

Decken die Übungen den bisher gelernten Stoff ab und sind sie abwechslungsreich formuliert?

**Die Lösungen zu den Übungen sind angegeben**

Enthält der Kurs die Lösungen zu allen angebotenen Fragen und Programmieraufgaben?

**Die Schwierigkeitsgrade der Übungen sind angegeben**

Enthalten die Fragestellungen Hinweise auf die Komplexität der jeweiligen Aufgabe?

**Die Lektionen enthalten am Ende eine Zusammenfassung der gelernten Konzepte**

Werden am Ende einer Lektion noch einmal die gelernten Konzepte in einer abschließenden Zusammenfassung erwähnt?

### 5.1.3 Kriterien zur Präsentation

Die Präsentation umfasst das Layout sowie die Darstellung der zu lernenden Inhalte. Die 17 hier aufgeführten Kriterien dienen der Bewertung dieser Aspekte.

**Der Inhalt ist in einzelne Lektionen (Kapitel) aufgeteilt**

Besitzt der Kurs eine Struktur aus aufeinander aufbauenden Kapiteln?

**Einzelne Lektionen sind überschaubar**

Sind die Inhalte der Lektionen klein genug, um während des Arbeitens den Überblick zu behalten?

**Die Lernziele sind am Anfang der Lektionen klar formuliert**

Enthalten die einzelnen Lektionen eine einleitende Beschreibung der vermittelten Konzepte?

**Teile, die sich auf andere JDK-Versionen beziehen sind kenntlich gemacht**

Sind Unterschiede in der Implementierung zwischen den verschiedenen JDK-Versionen („Java Development Kit“) entsprechend markiert?

**Motivierender Schreibstil**

Ist der Schreibstil der Zielgruppe angemessen? Wird der Leser zum Weiterlesen angeregt?

**Hervorhebung wichtiger Begriffe im Text**

Werden im Text wichtige Schlüsselbegriffe hervorgehoben?

**Der Text enthält Links zu anderen Lektionen**

Enthält der Text Hyperlinks zu anderen Teilen im Kurs?

**Der Text enthält Links zu externen Seiten**

Enthält der Text Hyperlinks zu externen Ressourcen im WWW?

**Hyperlinks werden überlegt eingesetzt**

Tauchen die Links an vernünftiger Stelle im Text auf? Sind die Texte der Links aussagekräftig?

**Hervorhebung der verschiedenen Linkarten (intern/extern)**

Kann der Leser anhand des Aussehens eines Hyperlinks die verschiedenen Linkarten (z. B. durch kleine Piktogramme) unterscheiden?

**Verwendung von Grafiken und Diagrammen zur Verdeutlichung**

Werden im Text bei Bedarf Grafiken, Fluss- oder UML-Diagramme eingesetzt, um komplexere Vorgänge zu verdeutlichen?

**Verwendung von weiteren multimedialen Inhalten**

Werden außer Grafiken noch andere Multimedia-Komponenten (Video, Ton und Animationen) verwendet?

**Ansprechendes und übersichtliches Layout**

Ist das gesamte Erscheinungsbild des Kurses ansprechend? Werden Farben, Grafiken und Frames vernünftig eingesetzt, ohne den Leser zu verwirren oder abzulenken?

**Syntax-Highlighting bei Code Snippets**

Wird Syntax-Highlighting (farbliche Hervorhebung von Schlüsselwörtern und Kommentaren) eingesetzt, um die Lesbarkeit der Code-Beispiele zu verbessern?

**Farbliche Hervorhebung von Beispielen, Tipps o.ä. (dezent)**

Sind besondere Bemerkungen wie Syntax-Beschreibungen, Warnungen und Tipps besonders vom umgebenden Text abgehoben?

**Hervorhebung ist im gesamten Kurs konsistent**

Ist diese Hervorhebung in allen Lektionen einheitlich?

**Tippfehler, Fehler im HTML-Code**

Enthält der Kurs Fehler (Fehler im Text oder im HTML-Code, z. B. „broken links“)?

### 5.1.4 Kriterien zur Bedienung

Die neun Kriterien zur Bedienung beinhalten die Führung des Benutzers sowie allgemein die Anwenderfreundlichkeit und Komplexität der Oberfläche.

**Es existiert eine klare Führung des Studenten („Next lesson“-Link)**

Kann der Benutzer über einen eigenen Hyperlink direkt zur nächsten Lerneinheit springen?

**Leichte Orientierung innerhalb des gesamten Kurses**

Weiß der Leser stets, an welcher Stelle er sich im Kurs befindet? Ist die Navigation innerhalb der aktuellen Lektion und zu anderen Lektionen leicht und übersichtlich?

**Schnelle Ladezeit (kleine Files)**

Sind die einzelnen Dateien des Kurses klein, sodass bei Nutzung über ein Netzwerk der Leser keine langen Wartezeiten hat?

**Vernünftige Reduzierung der Menge an Multimedia-Daten**

Sind die multimedialen Elemente des Kurses angemessen komprimiert (JPEG, MPEG o. ä.)?

**Glossar vorhanden**

Enthält der Kurs ein Glossar zur Klärung wichtiger Fachtermini?

**Glossar ist verlinkt**

Sind die Textstellen von den Glossareinträgen aus über Hyperlinks erreichbar?

**Stichwortindex vorhanden**

Enthält der Kurs ein Stichwortindex, um direkt auf spezielle Informationen zugreifen zu können?

**Stichwortindex ist verlinkt**

Sind die Einträge im Stichwortindex mit den entsprechenden Textstellen über Hyperlinks verknüpft?

**Volltextsuche vorhanden**

Verfügt der Kurs über eine Volltextsuche aller Lektionen? Wie ausgereift sind die Suchmöglichkeiten (Unterstützung von Wildcards, regulären Ausdrücken o. ä.).

## 5.2 Ergebnis der Evaluation

Anhand der beschriebenen Kriterien ließen sich die in Tabelle 5.1 auf Seite 38 angegebenen Java-Kurse und Tutorials auf ihren Einsatz als Grundlage für das in dieser Arbeit vorgestellte adaptive Lehrsystem überprüfen (s. Anhang A). Bei der abschließenden Benotung wurden jeweils die drei Bereiche *Inhalt*, *Präsentation* und *Erforderliche Anpassungen* anhand von Schulnoten („1: Sehr gut“ bis „6: Ungenügend“) bewertet.<sup>1</sup>

### 5.2.1 Inhalt

Die Evaluation zeigte wie zu erwarten enorme Unterschiede sowohl in der inhaltlichen Vollendung als auch in der Präsentationsform. Bei dem eigentlichen Kursinhalt sind einige der betrachteten Kurse schlicht zu knapp in ihren Ausführungen oder behandeln den theoretischen Hintergrund zu oberflächlich, sodass kein umfassender Lernerfolg garantiert ist. Die Kurse mit sehr großem Stoffumfang, die auch die theoretischen Hintergründe angemessen abdecken, sind Online-Versionen von gedruckten Büchern [17], [33].

---

<sup>1</sup>Diese Benotung ist natürlich durchaus subjektiv, und man sollte sich als Autor eines der getesteten Kurse in keiner Weise angegriffen fühlen.

### 5.2.2 Präsentation

Die Spanne bei der Präsentationsform reicht von Kursen, die aus einfachen nach HTML konvertierten Textdateien bestehen und die Möglichkeiten des WWW gar nicht ausschöpfen (weder Einsatz von Multimedia noch von Verweisen) bis hin zu benutzerfreundlichen und stilistisch gut organisierten hypermedialen Kursen.

Während einige einfach zu realisierende Merkmale wie Aufteilung des Kurses in einzelne Kapitel und einfache Navigationslinks zur nächsten empfohlenen Lektion in allen getesteten Kursen zu finden sind, sind andere Eigenschaften, die eine besondere Anpassung an den Einsatz im World Wide Web darstellen (beispielsweise die Möglichkeit zur Volltextsuche im gesamten Kurs und Verweise auf externe Ressourcen im Internet) nur vereinzelt anzutreffen. Ursache dafür dürfte die Tatsache sein, dass einige der Kurse nicht primär für den Einsatz im World Wide Web entwickelt wurden, sondern beispielsweise aus Vorlesungsnotizen entstanden sind.

### 5.2.3 Erforderliche Anpassungen

Die erforderlichen Anpassungen für den Einsatz als Inhalt eines adaptiven Lehrsystems schwankt ähnlich stark wie die beiden bereits beschriebenen Bereiche. Diese Anpassungen reichen von der Ergänzung eines Kurses um Übungen und Tests über Verbesserungen des Layouts bis hin zu einer kompletten Neustrukturierung des Inhalts. Um den technischen Aufwand möglichst gering zu halten, wurde bei der Benotung dieses Bereichs vorwiegend die einfache Eingliederung des Kursinhalts in das System iTeach bewertet.

Anhand der Evaluationskriterien wurden schließlich mit deutlichem Abstand die folgenden drei Kurse als sehr gut geeignet bewertet:

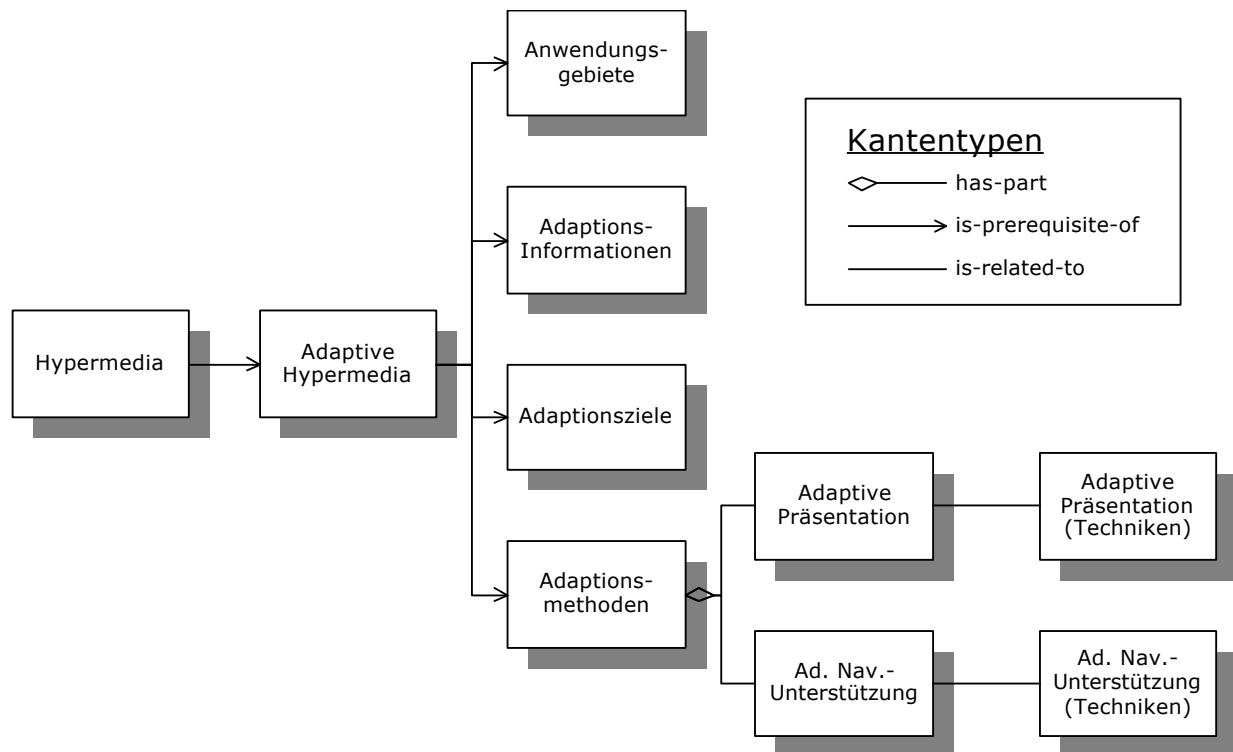
- Das *Java Tutorial* von Sun Microsystems
- *Thinking in Java* von Bruce Eckels
- *Goto Java 2* von Guido Krüger

Für den Einsatz als praktikumsbegleitendes Lehrmaterial eignet sich im hier vorgesehenen Kontext ein Kurs in deutscher Sprache, sodass als „Testsieger“ der sehr umfangreiche Kurs von Guido Krüger hervorging.

## 5.3 Adaptive Hypermedia

Da die Erstellung des Inhalts für das Java Online-Praktikum noch nicht abgeschlossen ist, wurde Kapitel 3 der vorliegenden schriftlichen Ausarbeitung als Kursinhalt verwendet, um das in den folgenden Kapiteln beschriebene System iTeach im praktischen Umfeld testen zu können, wobei der eigentliche Inhalt um einfache Übungsaufgaben zur Vertiefung des vermittelten Stoffs erweitert wurde.

Abbildung 5.1 zeigt das in diesem Zusammenhang verwendete Domänenmodell als semantisches Netz. Dieser Graph wird in einer XML-Datei spezifiziert, deren Format in Anhang B beschrieben wird.



**Abbildung 5.1:** Das im Rahmen des Kurses „Einführung in Adaptive Hypermedia“ verwendete Domänenmodell. Die Knoten repräsentieren die Konzepte, die Kanten die unterschiedlichen Beziehungen zwischen den einzelnen Konzepten.

Im folgenden Kapitel werden neben dem Domänenmodell auch die anderen Komponenten eines adaptiven Lehrsystems auf der Basis mathematischer Formalismen beschrieben.

# Kapitel 6

## Adaptivität in iTeach

Das im Rahmen dieser Arbeit implementierte System iTeach ist ein Lehrsystem, das sich individuell an den Benutzer und seinen Kenntnisstand über die vermittelte Domäne anzupassen vermag. Grundlage dafür sind – wie in adaptiven Lehrsystemen üblich und erforderlich und in Kapitel 4 bereits im allgemeinen Rahmen erläutert – neben Modellen der Domäne und des Benutzers eine Inferenz- und eine Adaptionskomponente, die für die Aktualisierung des Benutzermodells bei Interaktionen mit dem System bzw. für die individuelle Anpassung des Systems zuständig sind.

Dieses Kapitel stellt diese vier Komponenten von iTeach vor, indem die Begriffe der iTeach-Domäne und des iTeach-Benutzers formal definiert werden. Auf dieser Grundlage werden anschließend die Adaption- und Inferenzkomponente und die implementierten Algorithmen erläutert. Die Definitionen und Algorithmen stellen eine Beschreibung der Funktionsweise des Systems dar, die in dieser Form praktisch direkt im Code wiederzufinden sind, sodass dieses Kapitel gut zur Einarbeitung in die interne Arbeitsweise von iTeach eingesetzt werden kann.

### 6.1 Das Domänenmodell

Das Domänenmodell von iTeach besteht aus zwei Schichten: Auf der abstrakten Ebene sind Konzepte und ihre gegenseitigen Abhängigkeiten als semantisches Netz (Konzeptgraph) modelliert, wie dies auch in anderen Systemen der Fall ist (vgl. Abschnitt 4.5).

Auf einer niedrigeren Ebene sind einzelne atomare Fragmente definiert, die die verschiedenen Konzepte erklären. Jedes einzelne Fragment kann mehrere Konzepte zu einem gewissen Grad erklären; es existiert jedoch ein ausgezeichnetes „Hauptkonzept“, was die gesamte Modellierung und Implementierung erheblich vereinfacht. Abbildung 6.1 verdeutlicht das Zusammenspiel der beiden verschiedenen Modellierungsebenen. Im folgenden werden diese beiden Schichten formal beschrieben, um eine klar definierte Basis für die Beschreibung der Inferenz-Algorithmen zu erhalten.

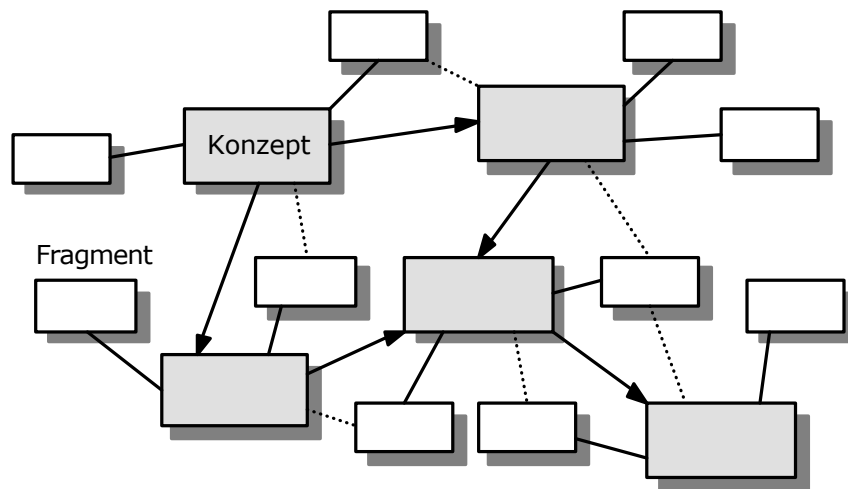


Abbildung 6.1: Die zwei Schichten des Domänenmodells in iTeach.

### 6.1.1 Abstrakter Level: Die Konzepte

Zur Beschreibung der abstrakten Schicht, die aus einem semantischen Netz bzw. Konzeptgraphen besteht, sind einige Hilfs-Definitionen erforderlich, die die Strukturelemente dieses Graphen erklären.

DEFINITION 6.1 Ein **Konzept-Knoten**  $v$  ist ein Quadrupel  $v = (id, n_v, s, F)$  mit den Eigenschaften:

1.  $id$  ist eine domänenweit eindeutige Zeichenkette.
2.  $n_v$  ist der Name des Konzepts.
3.  $s$  ist ein boolescher Wert, der angibt, ob der Knoten ein direkt lernbares Startkonzept repräsentiert.
4.  $F$  ist eine nicht-leere, aufsteigend topologisch sortierte Folge aller Fragmente, für die  $v$  Hauptkonzept ist (vgl. Definition 6.10).

Aus Gründen der Einfachheit werden im Folgenden die Begriffe Knoten und Konzept synonym gebraucht. Die Konzeptknoten sind über Kanten unterschiedlicher Kantentypen miteinander verbunden:

DEFINITION 6.2 Ein **Kantentyp**  $t$  ist ein Tupel  $t = (n_t, req)$  mit den Eigenschaften:

1.  $n_t$  ist eine domänenweit eindeutige Typbezeichnung.
2.  $req$  ist ein boolescher Wert, der angibt, ob das Startkonzept von Kanten dieses Typs eine erforderliche Vorbedingung des Zielkonzepts darstellt (s. Definition 6.5).

Hiermit können nun Kanten zwischen Konzepten definiert werden:

DEFINITION 6.3 Eine gerichtete **Kante**  $e$  zwischen zwei Konzepten  $q$  und  $z$  ist ein Quintupel  $e = (q, z, t, p, s)$  mit den folgenden Eigenschaften:

1.  $q$  ist das Startkonzept.
2.  $z$  ist das Zielkonzept.
3.  $t$  ist der Kantentyp.
4.  $p$  ist eine ganzzahlige Bewertung (Priorität).
5.  $s$  ist eine ganzzahlige Bewertung (Schwellwert). Es muss gelten:  $0 \leq s \leq k_{\max}$ , wobei  $k_{\max}$  den größtmöglichen Wert für den Wissensstand eines Konzepts bezeichnet (s. Definition 6.4).

Mit diesen Definitionen kann nun der Konzeptgraph eingeführt werden.

DEFINITION 6.4 Ein **Konzeptgraph**  $G$  ist ein Quadrupel  $G = (V, T, E, k_{\max})$  mit den Eigenschaften:

1.  $V$  ist eine nicht-leere Menge von Konzept-Knoten
2.  $T$  ist eine Menge von Kantentypen
3.  $E$  ist eine Menge von gerichteten Inter-Konzept-Kanten
4.  $k_{\max} \in \mathbb{Z}_+ \setminus \{0\}$  ist eine domänenweite Konstante, die das maximal erreichbare Wissen über ein Konzept angibt.

Im folgenden sei stets ein Konzeptgraph  $G = (V, T, E, k_{\max})$  definiert.

DEFINITION 6.5 Seien  $q, v \in V$ .

Die **Menge aller eingehenden Kanten** von  $v$  ist definiert als

$$E_{v,\text{in}} := \{e \in E \mid e \text{ hat Zielkonzept } v\}.$$

Analog ist die **Menge aller ausgehenden Kanten** von  $v$  definiert als

$$E_{v,\text{out}} := \{e \in E \mid e \text{ hat Startkonzept } v\}.$$

$q$  heißt (**erforderliche**) **Vorbedingung** für  $v$  genau dann, wenn gilt:

$$\exists e \in E_{v,\text{in}} : e = (q, v, t, p, s) \text{ mit } t = (n_t, \text{true}) \text{ (in Zeichen: } q \prec v).$$

Analog heißt  $q$  **nicht erforderlich** für  $v$  genau dann, wenn gilt:

$$\forall e \in E_{v,\text{in}} : e = (q, v, t, p, s) \text{ mit } t = (n_t, \text{false}) \text{ (in Zeichen: } q \not\prec v).$$

$v$  heißt **Folgekonzept** von  $q$  genau dann, wenn gilt:  $E_{q,\text{out}} \cap E_{v,\text{in}} \neq \emptyset$ .

Ein Konzeptgraph kann also letztendlich als eine Spezialisierung eines gerichteten Graphen [36] aufgefasst werden.

Damit ein Konzeptgraph  $G = (V, T, E, k_{\max})$  sinnvoll eingesetzt werden kann, ist es notwendig, dass die folgenden Bedingungen erfüllt sind:

- Für jede Zusammenhangskomponente [36] von  $G$  existiert mindestens ein Konzept-Knoten  $v' = (id, n_{v'}, s, F) \in V$  mit  $s = \text{true} \vee E_{v',in} = \emptyset$ . Anderenfalls können Konzepte vom Inferenzalgorithmus ignoriert werden.
- Es existieren keine parallelen Kanten, d. h. für alle paarweise verschiedenen Konzepte  $v_1, v_2 \in V$  gilt:  
 $0 \leq |E_{v_1,out} \cap E_{v_2,in}| \leq 1$ . Anderenfalls ist unklar, welche der Prioritäten und Schwellwerte verwendet werden.

Der Konzeptgraph wird entsprechend dieser Definitionen in einer XML-Datei definiert, deren genaue Syntax in Anhang B beschrieben wird.

### 6.1.2 Konkreter Level: Die Fragmente

Zur Beschreibung dieses Levels mit seinen eigentlichen Komponenten, den Fragmenten, sind einige Hilfsdefinitionen erforderlich, die in erster Linie die Metadaten der Fragmente betreffen; neben den hier betrachteten „wichtigen“ Metadaten Reihenfolgeconstraints, Anzeigeregeln und Nachbedingungen werden für jedes Fragment weitere „Begleiterscheinungen“ wie Autor, Erstellungsdatum und Beschreibung angegeben, die in der formalen Darstellung jedoch nicht berücksichtigt werden, da sie für die Beschreibung der Algorithmen nicht benötigt werden.

**DEFINITION 6.6** Ein **Reihenfolgeconstraint**  $oc$  ist ein Tripel  $oc = (f_1, f_2, t)$  mit den Eigenschaften:

1.  $f_1, f_2$  sind Fragmente mit gleichem Hauptkonzept (s. Definition 6.10)
2.  $t \in \{\text{before}, \text{after}\}$  ist der Typ, der die Reihenfolge von  $f_1$  bezüglich  $f_2$  festlegt

Anhand dieser Reihenfolgeconstraints lassen sich die Fragmente eines Konzepts topologisch sortieren; fehlen für zwei Fragmente Angaben zur Sortierung (d. h. die Fragmente nehmen nicht aufeinander Bezug), wird zufällig eine Reihenfolge festgelegt.

**DEFINITION 6.7** Eine **Bedingung**  $c$  ist ein Tupel  $c = (o, C)$  mit den Eigenschaften:

1.  $o \in \{\text{OR}, \text{AND}\}$  ist ein logischer Verknüpfungsoperator
2.  $C$  ist eine Menge von Einzelbedingungen, wobei zwischen lernweg- und wissensbasierten Einzelbedingungen unterschieden wird (s. Definitionen 6.20 und 6.21)

Ob eine Bedingung erfüllt ist, hängt vom individuellen Benutzer ab und wird erst dort näher beschrieben (s. Definition 6.22).

DEFINITION 6.8 Eine **Anzeigeregeln**  $pr$  ist ein Tripel  $pr = (c, a, p)$  mit den Eigenschaften:

1.  $c$  ist eine Bedingung
2.  $a \in \{\text{SHOW, SHRINK, HIDE}\}$  ist die **Aktion**; diese Aktion wird „ausgeführt“, also das Fragment entsprechend angezeigt, wenn  $c$  erfüllt ist (vgl. Definition 6.22).
3.  $p \in \mathbb{Z}_+$  ist eine Gewichtung, die **Priorität**

Die Anzeigeregeln können als eines der „Herzstücke“ des gesamten Systems angesehen werden; geben sie doch an, in welcher Form einem individuellen Benutzer ein Fragment präsentiert wird.

DEFINITION 6.9 Eine **Nachbedingung**  $pc$  ist definiert als ein Tripel  $pc = (v, c, k)$  mit den Eigenschaften:

1.  $v \in V$  ist ein Konzept
2.  $c$  ist eine Bedingung (diese ist optional, d. h. kann formal durch eine Bedingung ersetzt werden, die stets `true` liefert).
3.  $k$  mit  $0 < k \leq k_{\max}$  ist die relative Wertänderung von  $v$ , die bei Anzeige des Fragments im Wissensmodell des Benutzers eingetragen wird, vgl. Definitionen 6.14 und 6.19.

$pc$  heißt **erfüllt** genau dann, wenn  $c$  erfüllt ist (vgl. Definition 6.22).

Mit diesen Voraussetzungen kann nun ein vollständiges Fragment definiert werden, wobei zwischen Übungsaufgaben und Fragmenten an dieser Stelle nicht unterschieden werden muss:

DEFINITION 6.10 Ein **Fragment**  $f$  ist ein Quintupel  $f = (OC, PR, PC, t, v)$  mit den Eigenschaften:

1.  $OC$  ist eine Menge von Reihenfolgeconstraints
2.  $PR$  ist eine Menge von Anzeigeregeln
3.  $PC$  ist eine nicht-leere Menge von Nachbedingungen
4.  $t$  ist ein Titel, der im Falle einer SHRINK-Aktion angezeigt wird
5.  $v \in V$  ist ein ausgezeichnetes Konzept, das **Hauptkonzept** von  $f$

## 6.2 Das Benutzermodell

Auch für die Modellierung des individuellen Benutzers wurde die verbreitetste Technik verwendet – die Modellierung mithilfe eines Overlaymodells, das optional über eine Benutzer-Befragung und anschließende Stereotypenklassifizierung initialisiert werden kann. Durch eine solche Initialisierung und die anschließende Inferenz dienen ggf. neben den explizit ausgewiesenen Startknoten auch andere Knoten als „Einstiegspunkte“. Für jeden Benutzer werden eine Vielzahl von Daten gespeichert; diese umfassen neben „herkömmlichen“ Daten wie Login-Kennung und Passwort das Datum des ersten sowie letzten Logins. Der Übersichtlichkeit halber werden diese in der Definition des iTeach-Benutzers nicht mit angeführt, da sie für die Beschreibung der Adaptions- und Inferenzkomponente nicht benötigt werden.

Auch für die formale Beschreibung des Benutzermodells sind wieder mehrere Hilfsdefinitionen erforderlich. Zunächst Definitionen, die das Wissensmodell eines Benutzers betreffen; wir benötigen zwei ausgezeichnete Zustände (Konstanten), die ein Konzept im Wissensmodell eines Benutzers annehmen kann, wenn kein „echter“ Wissensstand verfügbar ist:

DEFINITION 6.11  $k_r$  ist ein ganzzahliger Wert mit  $k_r < 0$  ( $r$  steht dabei für „ready“).

$k_{nr}$  ist ein ganzzahliger Wert mit  $k_{nr} < 0$  und  $k_{nr} \neq k_r$  ( $nr$  steht dabei für „not ready“).

Die Menge der **möglichen Wissensstände** sei definiert als  $KV := \{k_{nr}, k_r, 0, \dots, k_{max}\} \subset \mathbb{Z}$ .

Wir können nun den Zusammenhang zwischen Konzepten und ihrem Wissensstand definieren:

DEFINITION 6.12 Die Funktion  $k : V \rightarrow KV$  ordnet jedem Konzept  $v \in V$  einen ganzzahligen Wert  $k(v) \in KV$ , den **Wissensstand**, zu.

DEFINITION 6.13 Ein Konzept  $v \in V$  heißt **lernbar** genau dann, wenn gilt:

$\forall e = (q, v, t, p, s) \in E_{v,in} : (q \not\prec v) \vee (q \prec v \wedge k(q) \geq s)$ . In diesem Fall gilt:  $k(v) \in KV \setminus \{k_{nr}\}$ .

Entsprechend heißt  $v$  **nicht lernbar** genau dann, wenn gilt:

$\exists e = (q, v, t, p, s) \in E_{v,in} : q \prec v \wedge k(q) < s$ . In diesem Fall gilt:  $k(v) = k_{nr}$ .

Das Wissensmodell eines Benutzers besteht aus einem Mapping aller Konzepte zu je einem Wissensstand:

DEFINITION 6.14 Ein **Wissensmodell**  $K$  bezüglich eines Konzeptgraphen  $G = (V, T, E, k_{max})$  ist definiert als Menge von Tupeln  $K = \{(v, k(v))\}$  mit den Eigenschaften:

1.  $\forall (v, k(v)) \in K : v \in V$  und  $k(v)$  ist der Wissensstand von  $v$ .
2. Für alle Tupel  $(v_i, k(v_i)), (v_j, k(v_j)) \in K$  mit  $1 \leq i, j \leq |V|, i \neq j$  gilt:  $v_i, v_j$  sind paarweise verschieden.

Der Einfachheit halber bezeichne im folgenden der Ausdruck  $K.v$  den Wissensstand des Konzepts  $v$  des Wissensmodells  $K$ . Es folgt noch eine hilfreiche Definition zur Aussage über den Wissensstand von Vorbedingungen:

DEFINITION 6.15 *Bezüglich eines Wissensmodells  $K$  heißt eine erforderliche Vorbedingung  $q$  eines Konzepts  $z$  **erfüllt** genau dann, wenn gilt:*

$\exists e = (q, z, t, p, s) \in E_{q,\text{out}} \cap E_{z,\text{in}}$  mit  $K.q \geq s$ .

Als nächstes folgen Definitionen zur History des Benutzers; sie besteht aus einer Menge von Fragmentzugriffen, wobei zwischen Fragmenten und Übungsaufgaben dahingehend unterschieden wird, dass für Aufgaben zusätzlich die Bearbeitungsdauer und der Hilfe-Level gespeichert wird.

DEFINITION 6.16 *Ein **Fragmentzugriff**  $fz$  ist definiert als ein Tripel  $fz = (f, t, a)$  mit den Eigenschaften:*

1.  $f$  ist ein Fragment
2.  $t$  ist die Zugriffszeit auf das Fragment, also ein Zeitstempel
3.  $a$  ist die Aktion, die angibt, wie  $f$  präsentiert wurde

DEFINITION 6.17 *Ein **Aufgabenzugriff**  $az$  ist definiert als ein Hextupel  $az = (f, t, a, p, h, r)$  mit den Eigenschaften:*

1.  $f, t, a$  wie für Fragmentzugriff
2.  $p$  ist eine Zeitspanne, die **Bearbeitungszeit**
3.  $h$  ist der maximale angeforderte Hilfelevel
4.  $r$  ist das Testergebnis in %.

DEFINITION 6.18 *Eine **iTeach-History**  $H$  ist eine nach der Zugriffszeit aufsteigend sortierte Folge von Fragment-Zugriffen:*

$H := \{z \mid z \text{ ist Fragment- oder Aufgabenzugriff}\}$ . Dabei wird je Fragment nur der letzte Zugriff gespeichert.

DEFINITION 6.19 *Ein **iTeach-User**  $U$  bezüglich eines Konzeptgraphen  $G = (V, T, E, k_{\max})$  ist ein Quintupel  $U = (c, n, g, K, H)$  mit den folgenden Eigenschaften:*

1.  $c, n, g \in V$  sind ausgezeichnete Konzepte:  $c$  das momentan bearbeitete,  $n$  das nächste empfohlene Konzept und  $g$  das momentan verfolgte Lernziel.
2.  $K$  ist das Wissensmodell bezüglich  $G$ .
3.  $H$  ist die History des Benutzers.

Abschließend kann nun beschrieben werden, wann eine Bedingung erfüllt ist, was von einem Benutzer  $U = (c, n, g, K, H)$  abhängig ist. Dazu zunächst die bereits angekündigten Definitionen von Einzelbedingungen:

DEFINITION 6.20 Eine **wissensbasierte Einzelbedingung**  $c'$  ist ein Tripel  $(v, \text{comp}, k)$  mit den Eigenschaften:

1.  $v \in V$  ist ein Konzept
2.  $\text{comp} \in \{<, \leq, =, \geq, >, \neq\}$  ist ein Vergleichsoperator
3.  $k \in KV$  ist ein Wissensstand von  $v$

$c'$  ist **erfüllt** genau dann, wenn gilt:  $K.v \text{ comp } k$ .

DEFINITION 6.21 Eine **lernwegbasierte Einzelbedingung**  $c''$  besteht lediglich aus einem Konzept  $v$ . Sie ist **erfüllt** genau dann, wenn gilt:  $v = c$ .

Eine Bedingung besteht nach Definition 6.7 aus einem Operator und einer Liste von Einzelbedingungen; sie ist erfüllt wenn – je nach Operator – entweder eine oder alle Einzelbedingungen erfüllt sind.

DEFINITION 6.22 Eine Bedingung  $c = (o, C)$  ist **erfüllt** genau dann, wenn genau eine der folgenden Eigenschaften erfüllt ist:

1.  $o = \text{or} \wedge \exists c' \in C : c' \text{ ist erfüllt für } U$
2.  $o = \text{and} \wedge \forall c' \in C : c' \text{ ist erfüllt für } U$

Auf diese Weise ist neben einer „wissensgesteuerten“ auch eine „lernweggesteuerte“ Adaption möglich.

## 6.3 Adaptive Techniken

iTeach verwendet verschiedene adaptive Techniken, um eine benutzerspezifische individuelle Präsentation zu ermöglichen. Neben den beiden Techniken „Conditional Text“ und „Stretch Text“ aus dem Bereich der adaptiven Präsentation wurden zur Navigationsunterstützung die direkte Führung sowie Link-Annotationen umgesetzt (vgl. Abschnitt 3.6 ab Seite 17).

Diese Techniken erscheinen am Erfolg versprechendsten, wie aus den in Abschnitt 3.7 beschriebenen Untersuchungen bereits ansatzweise hervorgeht. Außer dem Ausblenden von Links, das aufgrund der zu starken Einschränkung der Navigationsfreiheit des Benutzers als nicht sinnvoll eingestuft wurde, sind die implementierten Techniken die mit Abstand am häufigsten in bestehenden Systemen umgesetzten Techniken, was neben den Evaluationen als Garant für eine effektive Adaptivität angesehen wurde.

Die Techniken zur adaptiven Präsentation basieren auf einer regelgesteuerten Anzeige der einzelnen Fragmente. Im folgenden wird die Umsetzung dieser Regeln erläutert, bevor auf die Annotation von Links eingegangen wird. Die Vorgehensweise der direkten Führung wird durch die Algorithmen 6.9 und 6.10 ab Seite 59 beschrieben.

### 6.3.1 Fragment- und Testauswahl

Eine von iTeach generierte Webseite enthält mehrere Fragmente, die ein gemeinsames Hauptkonzept erklären; wie viele Fragmente angezeigt werden, wird von einer „Page-Full“-Heuristik gesteuert, die durch Implementierung als Strategiemuster [22] leicht austauschbar ist. Die Default-Implementierung liefert als Seitenumfang stets 3 Fragmente<sup>1</sup>, was als sinnvolles Mittelmaß zwischen „zu wenig Information pro Seite“ und „zu viel Scrollen nötig“ erscheint.

Der Algorithmus CHOOSE-FRAGMENTS wählt die Fragmente aus, die auf einer Seite präsentiert werden:

---

#### ALGORITHMUS 6.1 (CHOOSE-FRAGMENTS)

---

**Input:**

$v = (id, n_v, s, F)$ : Das Konzept

$U = (c, n, g, K, H)$ : Der Benutzer

**Output:** Eine geordnete Liste von Fragmenten mit ihrer jeweiligen Anzeige-Aktion

```

1  result := ∅;
2  for i := 0 to (F.length - 1) {
3      f := F[i];
4      if ((f ∉ H) ∧ (PAGE-FULL(result) = false))
5          result.append(f, GET-ACTION(f, U));
6  }
7  if (result = ∅) {
8      for j := 0 to (H.length - 1) {
9          if (PAGE-FULL(result) = false) {
10             fz = (f, t, a) := H[j];
11             result.append(fz, GET-ACTION(fz, U));
12         }
13     }
14 }
15 MAKE-FIRST-FRAGMENT-VISIBLE(result);
16 return result;
```

---

Für das übergebene Konzept  $v$  wird die sortierte Liste  $F$  durchlaufen (2) und für jedes unbekannte Fragment (4) wird durch Auswertung seiner Anzeigeregeln die benutzer-spezifische Aktion (SHOW, SHRINK oder HIDE) bestimmt (5). Über globale Konfigurati-

<sup>1</sup>Insofern ist es etwas verwegen, das Ganze als Heuristik zu bezeichnen ...

onsparameter kann dabei angegeben werden, ob Conditional-Text- und/oder Stretch-Text-Regeln ignoriert werden sollen. Nacheinander werden auf diese Weise solange Fragmente zur Ergebnisliste hinzugefügt, bis die PAGE-FULL-Heuristik angibt, dass die Seite voll genug ist. Durch die topologische Sortierung von F wird garantiert, dass nur Fragmente ausgewählt werden, deren Vorbedingungen erfüllt sind.

Wird auf diese Weise kein Fragment gefunden (7), d. h. alle Fragmente wurden bereits gelesen, werden unter erneuter Berücksichtigung der PAGE-FULL-Heuristik die ältesten Fragmente ausgewählt (8-12); da alle Fragmente bekannt sind, können die Reihenfolgeconstraints in diesem Fall ignoriert werden. Die Heuristik ist also eher als „obere Schranke“ zu verstehen, da evtl. weniger Fragmente als prinzipiell erlaubt sind auf einer Seite präsentiert werden.

Abschließend wird sichergestellt, dass mindestens ein Fragment (das erste) angezeigt wird (15) – auf diese Weise wird die Anzeige von Seiten vermieden, die nur geschrumpfte oder ausgeblendete Fragmente enthalten.

Die Auswahl einer Übungsaufgabe geschieht analog zu CHOOSE-FRAGMENTS mit dem Unterschied, dass genau eine Aufgabe präsentiert wird und die Aktion SHOW lautet.

### 6.3.2 Link-Annotationen

Die Annotationen der Links lassen sich in die beiden Gruppen adaptive und nicht-adaptive Annotationen aufteilen. Die Art der Annotationen ist jeweils ein kleines Icon vor dem Link, was an die Schriftgröße angepasst ist, sodass der Textfluss nicht zu sehr gestört wird. Unterschieden werden bei der Annotation von Konzepten die folgenden „Zustände“: Konzept besitzt noch unerfüllte Vorbedingungen (rotes Dreieck), Konzept ist lernbar, aber nicht empfohlen (gelbes Dreieck), Konzept wird empfohlen (grünes Dreieck) und Konzept wurde bereits gelernt (grüner Haken).

Analog wird bei Links zu Übungsaufgaben unterschieden: Verweise zu Aufgabenseiten von Konzepten, die hinreichend bekannt sind, dass Aufgaben prinzipiell gelöst werden können, sind mit einem gelben T (grün im Falle des empfohlenen Konzepts), anderenfalls mit einem roten T gekennzeichnet. Wurden alle Aufgaben zu einem Konzept bearbeitet, wird ebenfalls ein grüner Haken angezeigt.

Bei Fragmenten wird unterschieden, ob das Fragment bekannt ist (abgehakt) oder noch nicht (Papiersymbol). Alle erforderlichen Schranken und Bereichsangaben werden über globale Konfigurationsparameter eingestellt.

Nicht adaptiv werden Links zu externen Ressourcen (Rechteck mit Pfeil) und dem Glossar (Fragezeichen) markiert; ebenfalls kann die *Form* der Piktogramme (von den Häkchen abgesehen) als nicht adaptiv aufgefasst werden. So lassen sich allein durch die Form der Icons Links zu Konzepten, Fragmenten, Aufgaben, Glossareinträgen und externen Ressourcen unterscheiden; die Haken markieren einheitlich für Konzepte, Fragmente und Aufgaben bekannte Informationen.

## 6.4 Die Inferenzkomponente

Die Inferenzkomponente von iTeach ist neben der Aktualisierung des Wissensmodells eines Studenten für die Auswahl des Lernziels und des nächsten empfohlenen Konzepts zuständig. Die einzelnen Algorithmen werden im folgenden beschrieben.

### 6.4.1 Aktualisierung des Wissensmodells

Die Wissensmodellierung eines iTeach-Users ist nicht-monoton, d. h. das Wissen über Konzepte, Fragmente und Übungsaufgaben kann sowohl zu- als auch wieder abnehmen. Während die Wissenszunahme durch Präsentation von Informationen (Inhalte und Übungsaufgaben) erfolgt, kann die Wissensabnahme durch „automatisches“ Vergessen (nach Überschreitung verschiedener Time-out-Schwellwerte) wie auch durch die direkte Angabe negativer Werte als Nachbedingung eines Fragments erfolgen (was jedoch nur bedingt sinnvoll ist, da die Präsentation von Information selten zu einer Wissensabnahme führen dürfte).

Angaben zur Wissensänderung (sowohl Zu- als auch Abnahme) werden als *relative* Änderungen angegeben. Absolute Angaben werden nicht unterstützt, da zum einen die Semantik unpassend erscheint (durch Lesen eines Fragments mit absoluter Wissensänderung würde sämtliches Wissen über ein Konzept „überschrieben“, unabhängig vom aktuellen Wissensstand) und zum anderen die Präsentations-Reihenfolge der Fragmente Einfluss auf die Gesamtwissensänderung hat. Zur Verdeutlichung ein Beispiel:

Das System wählt drei Fragmente  $a$ ,  $b$  und  $c$  des gemeinsamen Hauptkonzepts  $v$  aus, um sie auf einer Seite zu präsentieren. Die Wissensänderungen betragen  $a := 3$  (absolut),  $b := +2$  und  $c := +3$  (beide relativ).

Sind die Fragmente voneinander unabhängig (d. h. es ist nicht durch Constraints eine feste Reihenfolge vorgegeben), so erhält  $v$  durch Anzeige von  $(a, b, c)$  den Wert 8; durch  $(b, c, a)$  jedoch den Wert 3.

Auch dieses Phänomen erscheint nicht realistisch, sodass aus diesen Gründen auf absolute Wertangaben in Nachbedingungen verzichtet wurde.

### Algorithmus zur Wissensaktualisierung

Der Algorithmus UPDATE-KNOWLEDGE aktualisiert das Wissensmodell eines Benutzers. Mögliche „Ausprägungen“ unterscheiden sich durch die Parameter, die übergeben werden: Wird eine Liste von Fragmenten und ihren zugehörigen Aktionen übergeben, läuft der Algorithmus wie erläutert ab; alternativ kann stattdessen auch direkt ein Mapping von Konzepten auf ihren neuen Wissensstand übergeben werden; in diesem Fall entfallen die Zeilen (1) und (2) und für die Aktualisierung der History ist der Aufrufer verantwortlich. Die dritte Möglichkeit besteht darin, eine Übungsaufgabe mit einem Ergebnis (in %) zu übergeben; in diesem Fall enthält  $F$  nur das Testobjekt und die Aktion SHOW, und das Mapping wird mit dem Testergebnis entsprechend verrechnet.

Die verwendeten Teilalgorithmen werden daran anschließend erläutert.

---

#### ALGORITHMUS 6.2 (UPDATE-KNOWLEDGE)

---

##### Input:

F: Eine Liste von Fragmenten mit ihren zugehörigen Aktionen

U = (c, n, g, K, H): Der Benutzer

##### Output: Das bisherige Wissensmodell des Benutzers

```

1  M := GET-KNOWLEDGE-CHANGE-MAP(F, U);
2  UPDATE-HISTORY(F, U);
3  M := REMOVE-TOO-OLD-FRAGMENTS(M, U);
4  R := K;
5  ∀m ∈ M {
6      if (M.m > K.m) {
7          K.m := M.m;
8          K := INC-BACKWARD-INFERENCE(K, m);
9          K := INC-FORWARD-INFERENCE(K, m);
10     } else if (M.m < K.m) {
11         K.m := M.m;
12         K := DEC-FORWARD-INFERENCE(K, m);
13     }
14  UPDATE-GOAL(K, g);
15  n := INFER-NEXT-CONCEPT(g);
16  return R;
```

---

Zunächst wird anhand der übergebenen Liste von Fragmenten und ihren Aktionen ein Mapping M aller betroffenen Konzepte auf ihre neuen Wissensstände ermittelt und die History aktualisiert (1–3, s. Algorithmen 6.3 bis 6.5).

In den Zeilen (5–13) findet die eigentliche Aktualisierung des Wissensmodells statt: Für jedes Konzept wird der Wert gesetzt, und die entsprechenden Inferenzalgorithmen werden ausgeführt (vgl. Algorithmen 6.6 – 6.8).

Abschließend wird das aktuelle Lernziel (14, Algorithmus 6.9) und darauf aufbauend das nächste empfohlene Konzept (15, Algorithmus 6.10) bestimmt.

#### Bestimmung der zu aktualisierenden Konzepte

Der Algorithmus GET-KNOWLEDGE-CHANGE-MAP liefert anhand der übergebenen Liste von Fragmenten ein Mapping aller betroffenen Konzepte auf ihren jeweiligen neuen Wissensstand.

---

#### ALGORITHMUS 6.3 (GET-KNOWLEDGE-CHANGE-MAP)

---

##### Input:

F: Eine Liste von Fragmenten mit ihren zugehörigen Aktionen

$U = (c, n, g, K, H)$ : Der Benutzer

**Output:** Ein Mapping aller durch  $F$  betroffenen Konzepte auf den jeweiligen neuen Wissensstand

Die Liste der übergebenen Fragmente und der jeweiligen Aktion wird durchlaufen und für jede zutreffende Nachbedingung eines Fragments, das nicht in der History enthalten ist, werden die betroffenen Konzepte in das Ergebnis-Mapping übernommen und die relative Wertänderung mit dem bisherigen Wissensstand verrechnet, wobei Mehrfachnennungen von Konzepten addiert werden. Aus Platzgründen wird auf eine formale Notation verzichtet; die Quellcode-Dokumentation enthält eine ausführlichere Beschreibung.

---

### Algorithmen zur Aktualisierung der History

Auf eine formale Beschreibung der Algorithmen UPDATE-HISTORY und REMOVE-TOO-OLD-FRAGMENTS wird ebenfalls verzichtet, die Umsetzung ist aber anhand des dokumentierten Quellcodes auch hier leicht nachzuvollziehen.

#### ALGORITHMUS 6.4 (UPDATE-HISTORY)

---

**Input:**

$F$ : Eine Liste von Fragmenten mit ihren zugehörigen Aktionen

$U = (c, n, g, K, H)$ : Der Benutzer

**Output:** Die aktualisierte History des Benutzers

Die übergebene Liste der Fragmente  $F$  wird durchlaufen und für jedes der enthaltenen Fragmente wird ein entsprechender Fragment- bzw. Testzugriff in der History gespeichert. Alte Einträge werden dabei überschrieben, sodass die History also zu jedem Fragment maximal einen Eintrag enthält.

---

#### ALGORITHMUS 6.5 (REMOVE-TOO-OLD-FRAGMENTS)

---

**Input:**

$M$ : Ein Mapping von Konzepten auf ihren Wissensstand

$U = (c, n, g, K, H)$ : Der Benutzer

**Output:** Das aktualisierte Mapping

Für jeden Eintrag der History wird die Zugriffszeit mit der aktuellen System-Zeit verglichen; wurde der globale Konfigurationsparameter `fragment-timeout` bzw. `test-timeout` (angegeben in Stunden) überschritten, wird der Eintrag gelöscht. Für jeden gelöschten Eintrag werden die betroffenen Konzepte und die jeweiligen Wissensänderungen ermittelt und mit den Einträgen in  $M$  verrechnet und schließlich das neue Mapping zurückgeliefert (s. Algorithmus 6.3).

---

### Rückwärts-Inferenz bei Wissenszunahme

Der Algorithmus INC-BACKWARD-INFERENCE beschreibt die Rückwärtsinferenz bei Wissenszunahme eines Konzepts  $v$ :

---

#### ALGORITHMUS 6.6 (INC-BACKWARD-INFERENCE)

---

**Input:**

$K$ : Das Wissensmodell des Benutzers

$v$ : Das aktualisierte Konzept

**Output:** Das aktualisierte Wissensmodell des Benutzers

```

1   $\forall e = (q, v, t, p, s) \in E_{v, in} \{$ 
2       $\text{if } ((q \prec v) \wedge (K.v \geq s) \wedge (K.q < s))$ 
3           $K.q := s;$ 
4   $\}$ 
5   $\text{return } K;$ 

```

---

Für jede eingehende Kante von  $v$  (1) wird der Wissensstand des Startkonzepts  $q$  auf den Schwellwert der entsprechenden Kante gesetzt (3), wenn  $q$  erforderliche Vorbedingung von  $v$  ist und der Schwellwert  $s$  überschritten wurde (2).

### Vorwärts-Inferenz bei Wissenszunahme

Die Wissenszunahme eines Konzepts  $v$  hat auch Einfluss auf die Folgekonzepte; der folgende Algorithmus aktualisiert das Wissensmodell des Benutzers entsprechend:

---

#### ALGORITHMUS 6.7 (INC-FORWARD-INFERENCE)

---

**Input:**

$K$ : Das Wissensmodell des Benutzers

$v$ : Das aktualisierte Konzept

**Output:** Das aktualisierte Wissensmodell des Benutzers

```

1   $\forall e = (v, z, t, p, s_e) \in E_{v, out} \{$ 
2       $\text{if } ((K.v \geq s_e) \wedge (K.z = k_{nr}) \wedge$ 
3           $(\forall f = (q, z, t, p, s_f) \in E_{z, in} \text{ mit } (q \neq v) \text{ gilt:}$ 
4               $(q \not\prec z) \vee ((q \prec z) \wedge (K.q \geq s_f))))$ 
5           $K.z := k_r;$ 
6   $\}$ 
7   $\text{return } K;$ 

```

---

Jedes Folgekonzept  $z$  von  $v$  mit bisher unerfüllten Vorbedingungen (2) wird lernbar (5), wenn alle erforderlichen Vorbedingungen von  $z$  erfüllt sind.

### Vorwärts-Inferenz bei Wissensabnahme

Nimmt das Wissen eines Konzepts  $v$  ab, so sind lediglich die Folgekonzepte betroffen, für die  $v$  erforderliche Vorbedingung ist:

---

#### ALGORITHMUS 6.8 (DEC-FORWARD-INFERENCE)

---

**Input:**

K: Das Wissensmodell des Benutzers

$v$ : Das aktualisierte Konzept

**Output:** Das aktualisierte Wissensmodell des Benutzers

```

1   $\forall e = (v, z, t, p, s) \in E_{v, out} \{$ 
2      if  $((v < z) \wedge (K.v < s))$ 
3           $K.z := k_{nr};$ 
4  }
5  return K;
```

---

Für jede ausgehende Kante von  $v$  (1) wird überprüft, ob  $v$  erforderliche Vorbedingung für das Zielkonzept ist und ob der Schwellwert unterschritten wurde (2); ist dies der Fall, wird das Zielkonzept der Kante auf nicht lernbar gesetzt.

### 6.4.2 Bestimmung des aktuellen Lernziels

Der Algorithmus UPDATE-GOAL zur Bestimmung des aktuell verfolgten Lernziels wählt unter allen unbekanntem Folgekonzepten bisher betrachteter Konzepte dasjenige mit der höchsten Priorität aus.

---

#### ALGORITHMUS 6.9 (UPDATE-GOAL)

---

**Input:**

K: Das Wissensmodell des Benutzers

$g$ : Das aktuelle Lernziel des Benutzers

$x$ : Wissensstand-Schwellwert, ab dem ein Konzept als gelernt betrachtet wird (globaler Konfigurations-Parameter)

**Output:** Das neue Lernziel des Benutzers.

---

```

1  if ((g ≠ null) ∧ (K.g < x)) return g;
2  M := {v | K.v ≥ 0};
3  if (M = ∅) {
4      S := {v | v ist Startkonzept};
5      if (S = ∅) return (zufälliges Konzept);
6      else return S.first();
7  } else {
8      newGoal := null; currentPriority := -1;
9      ∀m ∈ M {
10         ∀e = (m, z, t, p, s) ∈ Em,out {
11             if ((p > currentPriority) ∧ (K.z < 0)) {
12                 currentPriority := p; newGoal := z;
13             }
14         }
15     }
16     if (newGoal ≠ null) return (newGoal);
17     else {
18         for i := 0 to (kmax - 1) {
19             N := {v | K.v = i};
20             if (N ≠ ∅) return N.first();
21         }
22         return null;
23     }
24 }

```

---

Zu Beginn wird überprüft, ob ein neues Lernziel bestimmt werden muss; falls nicht, wird das bisherige Ziel als neues Ziel zurückgegeben (1). Zur Bestimmung eines neuen Ziels werden alle Konzepte bestimmt, die bereits betrachtet wurden (2); falls alle Konzepte unbekannt sind (3), wird – sofern verfügbar – ein Startkonzept zurückgegeben (6), ansonsten ein zufälliges Konzept (5).

Der interessante Fall wird ab (7) behandelt: Von allen unbekanntesten Folgekonzepten aller bekannten Konzepte wird das Konzept mit höchster Priorität bestimmt (9–15). Wird auf diese Weise ein Konzept gefunden, wird es zurückgegeben (16), falls nicht (d. h. alle Folgekonzepte sind bereits bekannt), wird eines der unbekanntesten Konzepte zurückgeliefert. Falls auch auf diese Weise kein Konzept bestimmt werden kann, sind alle Konzepte maximal bekannt und es wird als Spezialfall `null` zurückgegeben.

### 6.4.3 Auswahl des nächsten empfohlenen Konzepts

Das vom System vorgeschlagene nächste Konzept wird durch den rekursiven Algorithmus `INFER-NEXT-CONCEPT` bestimmt. Ausgehend vom übergebenen Konzept wird dazu jede noch unerfüllte Vorbedingung ihrerseits auf unerfüllte Vorbedingungen untersucht. Als nächstes Konzept wird schließlich das Konzept ausgewählt, das auf dem Pfad der größten Schwellwerte keine unerfüllten Vorbedingungen besitzt.

---

**ALGORITHMUS 6.10 (INFER-NEXT-CONCEPT)**

---

**Input:**

g: Das aktuell untersuchte Konzept

**Output:** Das nächste empfohlene Konzept des Benutzers

```
1  currentThreshold := 0; nextToCheck := null;
2   $\forall e = (q, g, t, p, s) \in E_{g, \text{in}} \{$ 
3      if  $((q \prec g) \wedge (K.q < s) \wedge (s > \text{currentThreshold}))$ 
4          currentThreshold := s; nextToCheck := q;
5      }
6  if (nextToCheck  $\neq$  null)
7      return INFER-NEXT-CONCEPT(nextToCheck);
8  else
9      return g;
```

---

In den beiden zuletzt vorgestellten Algorithmen werden die folgenden Heuristiken zur Konzept-Auswahl umgesetzt, wie dies auch in mindestens einem anderen System der Fall ist [27]:

- (Nahezu vollständig) bekannte Konzepte vermeiden
- Noch nicht lernbare Konzepte vermeiden
- Konzepte bevorzugen, die unmittelbar benötigt werden
- Konzepte bevorzugen, die eng mit schon bekannten Konzepten zusammenhängen
- Konzepte bevorzugen, deren Komplexität dem Wissensstand des Benutzers entspricht

Auf diese Weise liefert der dynamische „Weiter“-Link auf den generierten Webseiten eine Führung des Benutzers, die insbesondere für Benutzer mit geringem Metawissen über die Domänenstruktur von großer Hilfe ist. Das folgende Kapitel bietet einen Einblick in die allgemeine Systemarchitektur von iTeach; auf implementierungsnahe Details wird im daran anschließenden Kapitel eingegangen; aus Platzgründen sei aber zusätzlich auf die ausführliche Quellcode-Dokumentation verwiesen.

# Kapitel 7

## Einsatzmöglichkeiten von iTeach

Das System iTeach ist aufgrund seiner Architektur so konzipiert, dass es in verschiedenen Szenarios einsetzbar ist. Dieses Kapitel beschreibt diese verschiedenen Umgebungen und wie die jeweilige Integration von iTeach aussieht.

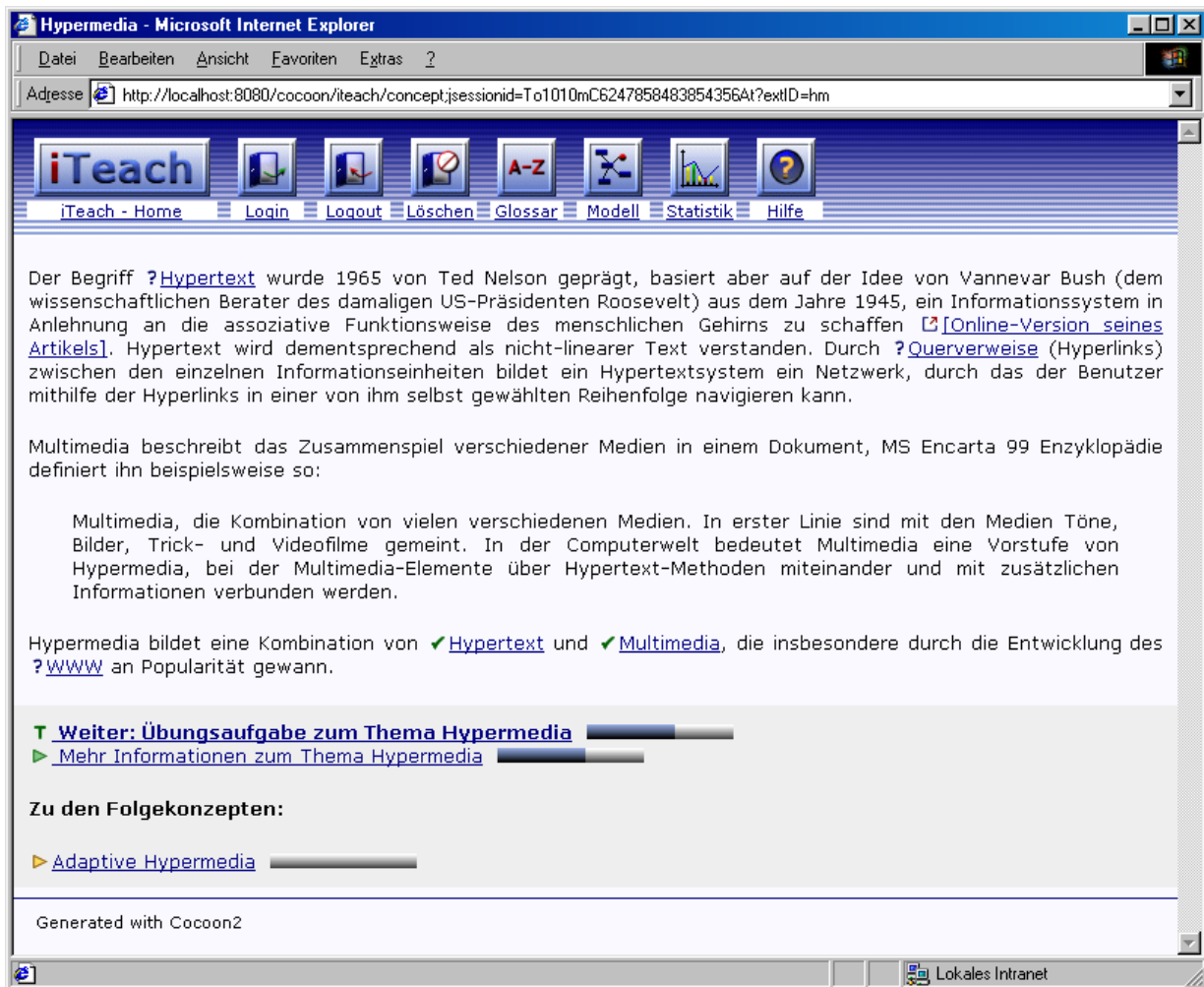
In erster Linie ist iTeach eine eigenständige Webapplikation; als Voraussetzungen sind lediglich ein Webserver mit Unterstützung von Java-Servlets (API 2.2) und Cocoon erforderlich (s. Anhang C für genauere Installationshinweise). Darauf aufbauend kann iTeach in Kombination mit dem am Lehrstuhl für Informatik VI entwickelten Informationssystem eingesetzt werden. Des Weiteren ist der Einsatz (insbesondere der Inferenzalgorithmen) im Rahmen des momentan noch in der Entwicklung befindlichen Java Online Praktikums am Lehrstuhl für Informatik II geplant, um eine intelligente Führung durch das Tutorial zu ermöglichen.

### 7.1 iTeach als eigenständige Webapplikation

iTeach bietet neben der eigentlichen Hauptkomponente, der Adaptivität, die Möglichkeit für den einzelnen Benutzer, sein gesamtes Wissensmodell einzusehen und zu manipulieren, sowie eine allgemeine Beurteilung seiner Lerneffizienz. Damit einher geht eine (verhältnismäßig rudimentäre) Benutzerverwaltung und mithilfe eines Glossars, dessen Einträge automatisch im Text erkannt werden, kann der Autor wichtige Begriffe an einer separaten Stelle zusammenstellen.

#### 7.1.1 Präsentation von Inhalt

Die Inhaltsseiten enthalten ein oder mehrere Fragmente zu einem bestimmten Konzept, die vom Algorithmus CHOOSE-FRAGMENTS (vgl. S. 53) bestimmt wurden. und zusätzlich eine Liste mit Links zu anderen Konzepten und Übungsaufgaben. Abbildung 7.1 zeigt eine Inhaltsseite, wie sie für einen Benutzer „ohne Wissen“ generiert wird und einige der Piktogramme, die zur Verdeutlichung der verschiedenen Linkarten eingesetzt werden.



**Abbildung 7.1:** Eine typische Inhaltsseite zum Konzept Hypermedia. Die Balkenanzeige hinter den Links am Ende der Seite gibt Aufschluss über den aktuellen Wissensstand des betreffenden Konzepts.

Die Liste mit Verweisen am Seitenende enthält Links zu den folgenden Zielen:

### Zurück, Vorwärts

Sind Verweise, die in der Browsing-History zurück bzw. vorwärts blättern; sie zeigen die vorige bzw. nächste Inhaltsseite an. Diese Links sind eine Erweiterung zu den Browser-eigenen Funktionen, da die URLs teilweise identisch sind und der Browser somit die älteren Inhalte überschreibt. Die Browsing-History enthält je nach Konfiguration die seit der ersten Registrierung oder – falls sie nicht persistent ist – nur die seit dem letzten Login angewählten Seiten.

### Weiter

Führt den Benutzer zur nächsten empfohlenen Seite, dies ist entweder eine

Inhalts- oder Aufgabenseite. Hinter diesem Link verbirgt sich letztlich das Ergebnis des INFER-NEXT-CONCEPT-Algorithmus (s. S. 61).

### Übungsaufgabe

Präsentiert eine passende Übungsaufgabe zum Konzept der aktuellen Seite. Dieser Link ist nur verfügbar, wenn der Weiter-Link zu einer Inhaltsseite führt und Tests für das Konzept vorhanden sind.

### Folgekonzepte

Eine Liste von Folgekonzepten des aktuellen Konzepts (dem Hauptkonzept der Fragmente auf der präsentierten Seite).

## 7.1.2 Präsentation von Übungsaufgaben

Übungsaufgaben werden im Gegensatz zu normalen Fragmenten stets einzeln auf einer Seite präsentiert, um die Bearbeitungsdauer jeder Aufgabe bestimmen zu können, die für die Beurteilung des Benutzers benötigt wird. Abbildungen 7.2 und 7.3 zeigen eine solche Übungsaufgabe zum Thema Hypertext und die vom System generierte Antwortseite.

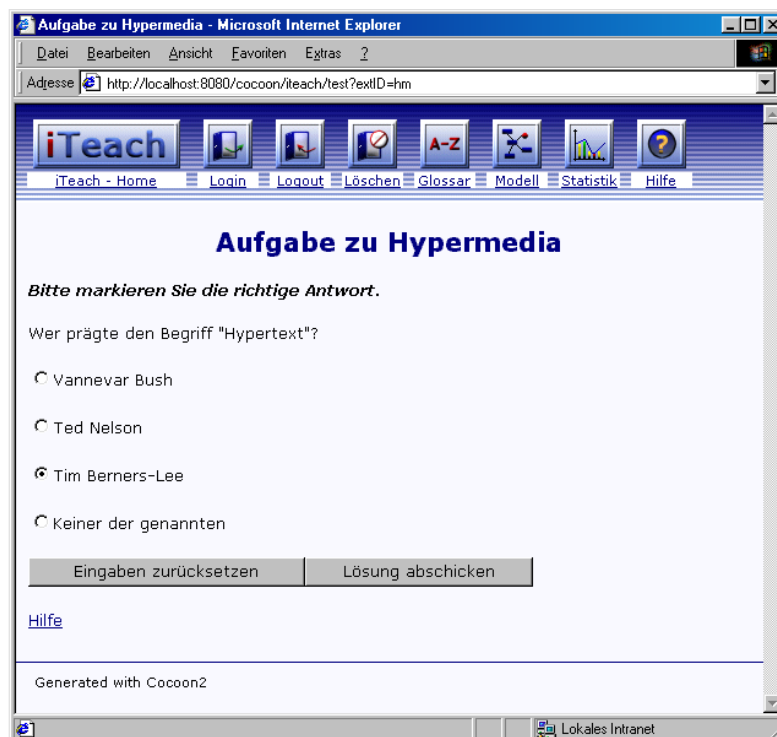
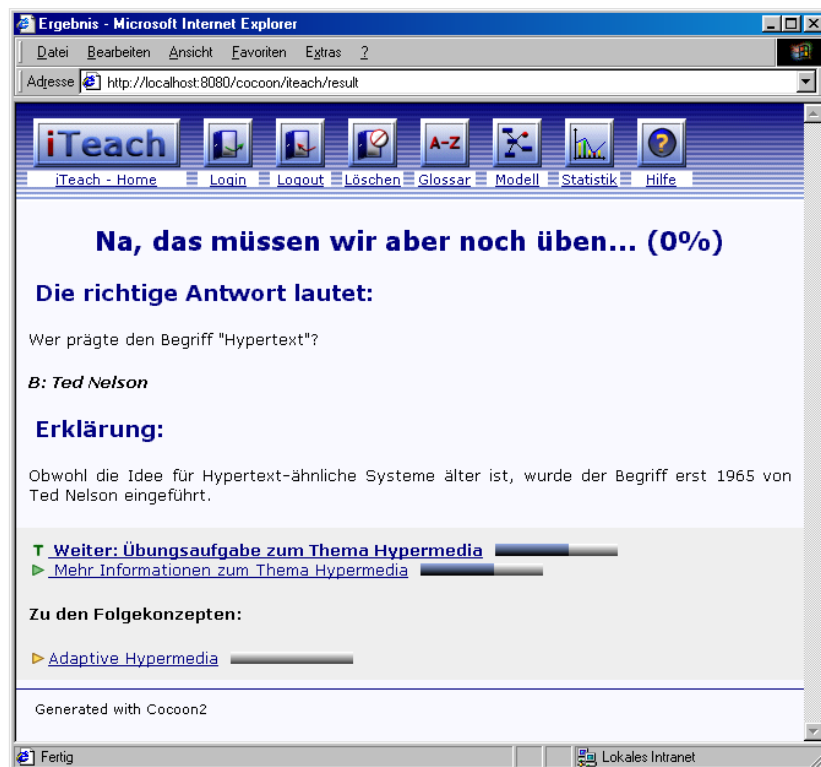


Abbildung 7.2: Eine MC-Übungsaufgabe zum Thema Hypermedia.



**Abbildung 7.3:** Die von iTeach generierte Antwortseite zur (falschen) Antwort „Tim Berners-Lee“ der Frage aus Abbildung 7.2. Aufgrund der falschen Antwort wird als nächste Seite eine weitere Übungsaufgabe zu Hypermedia empfohlen.

Über einen Hilfe-Link erhält der Benutzer auf der Aufgabenseite eine aufgaben-spezifische Hilfe. Für jeden Test können beliebig viele verschiedene Hilfestellungen angegeben werden, deren Auswahl über einen Timeout-Parameter gesteuert wird. Jede dieser Hilfestellungen verfügt ferner über einen „reduction value“, der angibt, um wie viele Prozentpunkte (nicht Prozent!), das Ergebnis bei Anzeige der jeweiligen Hilfe reduziert wird. Durch jeweiliges Setzen auf den Wert 0 kann dieses Verhalten deaktiviert werden.

Die Antwortseite enthält je nach Richtigkeit der Antwort eine entsprechende Beurteilung und im Falle einer nicht korrekten Antwort die Lösung sowie analog zu den Inhaltsseiten eine Linkliste, über die der Benutzer zu weiteren Aufgaben oder anderen Konzepten gelangen kann.

Bricht der Benutzer die Bearbeitung einer Aufgabe zu einem bestimmten Konzept ab, indem er einem Verweis innerhalb der Hilfe folgt, so gelangt er bei späterem Folgen zu einer Aufgabe desselben Konzepts auf jeden Fall wieder zu dieser Aufgabe, um sie fertig zu bearbeiten. Dabei läuft die „Stoppuhr“ weiter, bis die Aufgabe fertig bearbeitet wurde.

iTeach stellt Multiple-Choice- und Lückentext-Aufgaben zur Verfügung; durch die in Kapitel 8 geschilderte Implementierung lassen sich jedoch problemlos weitere Testty-

pen hinzufügen, wobei weder die vorhandenen Daten noch der Quellcode geändert werden müssen (lediglich die Konfigurationsdatei zur Angabe des Klassennamens und die betroffenen Stylesheets, vgl. Anhang B).

### 7.1.3 Weitere Funktionen

Die ständig sichtbare Kopfleiste (s. z. B. in Abbildung 7.2) ermöglicht den direkten Zugriff zu den folgenden Kommandos:

#### Registrierung (Auf der Login-Seite)

Um das System mehr als einmal zu verwenden, ist eine Registrierung erforderlich, sodass iTeach den individuellen Benutzer zu einem späteren Zeitpunkt erkennen kann. Neben Login-Kennung und Passwort ist der volle Name anzugeben, der jedoch lediglich „zur Anrede“ verwendet wird. Die registrierten Benutzer werden in einer XML-Datei gespeichert, wobei neben Loginkennung der MD5-Hash des Passworts gespeichert wird (als 32 Zeichen umfassender Hexadezimalstring). Für jeden Benutzer wird ferner eine separate XML-Datei mit den weiteren Daten wie Wissensmodell und History angelegt. Alle diese Dateien befinden sich im selben Verzeichnis, das aus Datenschutzgründen für den Zugriff von außen gesperrt werden sollte (dies kann in der Regel der Webserver übernehmen).

Nach erfolgreicher Registrierung (sie schlägt beispielsweise fehl, wenn die Einträge der beiden Passwortfelder nicht übereinstimmen, die Benutzerkennung bereits vergeben ist oder eines der Eingabefelder leer ist) wird der Benutzer zu einem „Fragebogen“ weitergeleitet, auf dem er sein Hintergrundwissen über die Domäne angeben kann (dieser Schritt kann bei entsprechender Konfiguration des Systems entfallen). Daraufhin wird das Wissensmodell mit den vom Autor festgelegten Werten initialisiert und der Benutzer gelangt automatisch zur ersten vom System vorgeschlagenen Inhaltsseite.

#### Login

Mithilfe des Login-Formulars kann sich ein Benutzer zu einem späteren Zeitpunkt erneut am System anmelden und wird automatisch zur nächsten vorgeschlagenen Inhaltsseite weitergeleitet.

Über einen Gast-Account (Kennung und Passwort `guest`) kann das System normal genutzt werden mit dem Unterschied, dass keine Daten langfristig gespeichert werden – die individuellen Daten wie Wissensmodell, Testergebnisse und History sind dementsprechend nur solange gültig, wie das System genutzt wird. Durch die nicht-persistente Speicherung der Daten in einem `Session`-Objekt, das vom Webserver bzw. Cocoon zur Verfügung gestellt wird, können beliebig viele Gast-Nutzer parallel mit dem System arbeiten.

#### Logout

Über einen Logout-Request wird das `Session`-Objekt gelöscht, wodurch im Falle eines Gast-Accounts auch die gespeicherten Daten gelöscht werden.

Vergisst ein Benutzer, sich vom System abzumelden, wird die Session in der Regel automatisch nach einem konfigurierbaren Zeitintervall vom Webserver gelöscht (bei Apaches Tomcat beträgt dieser Zeitraum defaultmäßig 30 Minuten).

### **Benutzerdaten löschen**

Um *alle* Benutzerdaten vom Server zu löschen, können über einen entsprechenden „unregister“-Request sowohl die eigentlichen benutzerspezifischen Daten als auch der entsprechende Eintrag aus der Liste der registrierten Benutzer gelöscht werden; schließlich wird auch die Session ungültig, sodass der Benutzer für die weitere Verwendung sich erneut registrieren oder als Gast anmelden muss. Für einen als Gast eingeloggten Benutzer ist die Wirkung analog zu einem regulären Logout.

### **Glossar**

Um den Autoren des vermittelten Inhalts zu ermöglichen, den Benutzern eine Liste mit Fachbegriffen o. ä. zur Verfügung zu stellen, kann eine separate Glossardatei verwendet werden. Unterstützt wird neben der Angabe von Synonymen die automatische Erkennung von Glossareinträgen im Text. Durch einen Konfigurationsparameter kann die Toleranz bezüglich Wortendungen gesteuert werden; er gibt die maximal erlaubte Anzahl Zeichen (ungleich Whitespace) am Wortende an, um einen Glossareintrag als solchen zu erkennen und vollständig als Link zu markieren. Auf diese Weise können Genitiv- und Pluralformen erkannt werden und der Textfluss wird nicht durch „halb“ verlinkte Wörter gestört. Alternativ dazu können Links auch explizit im Text angegeben werden.

### **Ansicht des Benutzermodells**

Jeder Benutzer kann sich seinen aktuellen Wissensstand anzeigen lassen und den Wissensstand der einzelnen Konzepte manipulieren. Über einen Update-Button wird das Modell mit anschließender Inferenz entsprechend aktualisiert. Neben dem Wissensstand wird für jedes Konzept die Zahl der gelesenen Fragmente sowie die Bearbeitung der Tests angezeigt. Abbildung 7.4 zeigt entsprechende Einträge für die Konzepte Hypermedia und Hypertext.

### **Benutzerstatistik**

Die Benutzer-Statistiken geben allgemeine Auskunft über die individuelle Lerneffizienz. Dazu werden neben der Bearbeitungsdauer der Übungsaufgaben auch der allgemeine Lernfortschritt und die bisher aufgewendete Zeit herangezogen. Aufgrund der recht hohen Ungenauigkeit von serverseitigen Zeitmessungen sollte der Benutzer JavaScript aktivieren, um die Messungen auf Clientseite durchführen zu können.

### **Online-Hilfe**

Die Online-Hilfe liefert eine kurze Beschreibung des Systems und eine Erklärung der einzelnen Funktionen und Piktogramme. Sie dient in erster Linie dazu, den Benutzer mit der Oberfläche vertraut zu machen.



**Abbildung 7.4:** Die Ansicht des Benutzermodells. Zu beachten ist, dass der Wissensstand von *Adaptive Methoden* 3 beträgt, jedoch noch keine Inhalte präsentiert wurden. Dieser Zustand kommt entweder durch eine entsprechende Stereotypen-Initialisierung des Wissensmodells oder eine manuelle Aktualisierung des Benutzermodells zustande.

## 7.2 iTeach im Informationssystem

Das momentan am Lehrstuhl für Informatik VI entwickelte Informationssystem<sup>1</sup> ist ein auf Java Servlets und einer relationalen Datenbank basierendes System zur Präsentation und Verwaltung von Informationen auf Basis unterschiedlicher Dokumenttypen (neben XML-Dokumenten werden auch PDF-Dateien und die Anbindung von Expertensystemen unterstützt) [52], [34].

<sup>1</sup>Z. B. unter <http://d3web.informatik.uni-wuerzburg.de:8666/d3web/servlet/Index>

### 7.2.1 Das Informationssystem

Die Dokumente sind hierarchisch organisiert und werden über eindeutige Kennungen identifiziert. In einem Inhaltsframe kann der Benutzer in einer Baumstruktur komfortabel den gesamten Inhalt einsehen und zu beliebigen Dokumenten navigieren [43]. Neben einer rollenbasierten Benutzerverwaltung stellt das System eine umfangreich konfigurierbare Volltextsuche [52] und das Hinzufügen von Bemerkungen zu einzelnen Seiten zur Verfügung [43]. Neben den IDs der Konzepte und Fragmente sind die Wortvektoren für die Volltextsuche in der Datenbank zu speichern.

Es sind besonders diese drei Eigenschaften – Inhaltsframe, Volltextsuche und Annotationen – die eine Integration von iTeach interessant machen.

### 7.2.2 Integration von iTeach

Die Integration von iTeach und dem Informationssystem basiert im Idealfall auf einer losen Kopplung beider Systeme; Requests an iTeach-Komponenten über die Oberfläche des Informationssystems (dem Inhaltsverzeichnis oder der Ergebnisseite einer Suchanfrage) könnten so vollständig an Cocoon und iTeach weitergereicht werden.

Prinzipiell kann mit Hilfe eines `RequestDispatcher`-Objekts das gesamte `Request`-Objekt an Cocoon weitergereicht werden; durch diese rein auf dem offiziellen Servlet API 2.2 basierende Kopplung sind beide Systeme jeweils allein einsatzfähig, und auch weit reichende Änderungen sind auf das jeweilige System beschränkt.

Probleme dieses an sich guten Ansatzes ergeben sich durch die Kombination der verschiedenen Systeme: Auf der einen Seite Webserver und Informationssystem, auf der anderen Cocoon und iTeach. Durch das Weiterleiten über einen `RequestDispatcher` werden jedoch vorhandene Session-Informationen gelöscht, sodass die für iTeach erforderlichen individuellen Informationen nicht vorhanden sind. Da das Problem vermutlich durch den eingesetzten Webserver (Tomcat 3.2) verursacht wird, sind die möglichen Alternativen der Einsatz eines anderen Servers (oder evtl. nur einer aktuelleren Version) oder eine festere Kopplung – diese Ansätze wurden jedoch im Rahmen dieser Arbeit nicht weiter verfolgt.

## 7.3 Java Online Praktikum

Eine weitere Umgebung, in der der Einsatz von iTeach geplant ist, ist das zur Zeit am Lehrstuhl für Informatik II entstehende Java Online Praktikum (JOP). Dieses System ist als entsprechender Kurs bei der Virtuellen Hochschule Bayern angemeldet worden und soll ab Mitte 2002 offiziell eingesetzt werden.

### 7.3.1 Das JOP-System

Die Architektur besteht aus der Kombination eines Java-Tutorials und eines „Praktomaten“, mit dessen Hilfe Programmieraufgaben und Beispiele gestellt und die abgegebenen Lösungen von sog. „elektronischen Korrektoren“ ausgewertet werden. Dies ist als Vorstufe zu den menschlichen Korrektoren zu verstehen, die letztlich für die Bewertung einer Lösung verantwortlich sind. Sowohl das Tutorial als auch der Praktomat sollen dabei prinzipiell eigenständig lauffähig sein.

### 7.3.2 Integration von iTeach

Das Java-Tutorial ist momentan Thema einer Diplomarbeit [20] und basiert auf Cocoon 2, sodass prinzipiell eine Kopplung von iTeach und dem Tutorial-System möglich ist; der Einsatz von iTeach wird (vermutlich) so aussehen, dass vornehmlich die Inferenzalgorithmen und die Wissensmodellierung des Benutzers eingesetzt werden können, um eine intelligente Führung durch den Kurs zu ermöglichen. Dabei soll das System jedoch auch ohne iTeach lauffähig sein, sodass bereits eine grundlegende Navigationsmöglichkeit (in Form von nicht-adaptiven Verweisen) zur Verfügung gestellt wird. Ferner enthält das System die Möglichkeit, „Workflows“ zu definieren (sowohl vom Autor als auch vom Benutzer selbst) und die einzelnen Fragmente zu sortieren – zwei Eigenschaften, die von iTeach in dieser Form nicht unterstützt werden, jedoch in einer Ausbaustufe neben anderen Fähigkeiten hinzugefügt werden können (vgl. Abschnitt 9.1.2).

Im folgenden Kapitel wird iTeach auf einer implementierungsnahen Schicht beschrieben; dabei werden, nach einigen Erläuterungen zur Implementierung an sich, die für die in diesem und im vorigen Kapitel beschriebene Funktionalität verantwortlichen Klassen und ihre Abhängigkeiten dargestellt.

# Kapitel 8

## Implementierung von iTeach

Bei dem im Rahmen dieser Arbeit entwickelte adaptiven Lehrsystem lag der Schwerpunkt bei der Realisierung neben der Umsetzung der in Kapitel 3 erläuterten adaptiven Techniken auf der leichten Erweiterbarkeit und Integrierbarkeit. Aus diesem Grund wurde versucht, den Code übersichtlich zu halten, d. h. mithilfe von Refactoring-Methoden selbsterklärend zu gestalten [21] und gut zu dokumentieren, um die Einarbeitungszeit in das System möglichst kurz zu halten.

Die vier Hauptkomponenten eines adaptiven Lehrsystems (vgl. Kap. 6) können aufgrund des im folgenden erläuterten modularen Aufbaus auf einfache Weise ausgetauscht werden.

### 8.1 Entscheidungen zur Implementierung

iTeach basiert auf zwei recht komplex anmutenden Frameworks – Avalon und Cocoon, beides Projekte der *Apache Software Foundation (ASF)*<sup>1</sup> und somit inklusive Sourcecode frei verfügbar. Der Vorteil, den diese Pakete bieten, ist ein gravierender Gewinn an Modularität, Wiederverwendbarkeit und Erweiterbarkeit des gesamten Systems, sodass die dadurch eventuell verursachte längere Einarbeitungszeit kaum ins Gewicht fällt.

Ferner wurde das JUnit Testing Framework<sup>2</sup> verwendet, um die Lauffähigkeit einzelner Code-Teile, insbesondere der in Abschnitt 6.4 beschriebenen Inferenzalgorithmen, zu gewährleisten und die Entwicklung zu beschleunigen.

Durch den ausschließlichen Einsatz von XML-Dateien zur persistenten Datenhaltung wurde die Abhängigkeit von eventuell plattformabhängigen Datenbanksystemen vermieden. Damit einhergehende mögliche Geschwindigkeitseinbußen wurden aufgrund der geringen Datenmengen vernachlässigt; für einen Einsatz mit vielen gleichzeitigen Zugriffen sollte jedoch ein intelligentes Caching der Benutzerdaten verwendet werden, um die Serverbelastung zu reduzieren.

---

<sup>1</sup><http://www.apache.org>

<sup>2</sup><http://www.junit.org>

### 8.1.1 Verwendete Entwurfsmuster

Um den Code weiter zu strukturieren, wurden die folgenden Entwurfsmuster [22] implementiert bzw. von Avalon und Cocoon übernommen:

#### Fabrikmethode

Sowohl Avalon als auch Cocoon verwenden u. a. das Muster „Fabrikmethode“, um einzelne Komponenten zu erzeugen – als Folge davon werden auch die Komponenten von iTeach zum Großteil mittels Fabrikmethoden erzeugt.

#### Singleton

Die Klasse `CourseManager`, das „Rückgrat“ des gesamten iTeach-Systems, wurde als Singleton implementiert, um sicherzustellen, dass maximal eine Instanz dieser Klasse existiert und mit ihm auch nur ein `Domain`-Objekt.

#### Strategie

Auf die Inferenzalgorithmen wird nur über eine Schnittstelle zugegriffen, wodurch diese Algorithmen ohne Änderungen des restlichen Systems ausgetauscht werden können. Ebenso wurde die „PageFull-Heuristik“ zur Feststellung, ob die generierte Seite „voll genug“ ist, als Strategiemuster implementiert.

#### Delegation

Die Auswertung der Übungsaufgaben wird an die jeweiligen Testobjekte delegiert, um einen rein schnittstellen-basierten Zugriff auf diese Objekte zu ermöglichen.

Des Weiteren wurden von Java bereits zur Verfügung gestellte Entwurfsmuster verwendet, u. a. *Iterator* im Rahmen des vielfach eingesetzten Collection-Frameworks im Paket `java.util` und *Immutable* bei Methoden, die ein `List`- oder `Map`-Objekt zurückliefern, dieses aber vor unkontrollierten Änderungen bewahrt werden soll.

Während Refactoring und Entwurfsmuster die eigentliche Implementierung beeinflussen, sind offizielle Standardisierungsbemühungen in das Format der persistenten Datenthaltung von iTeach eingegangen.

### 8.1.2 Berücksichtigte Standards

Als Basis der XML-Dateien und ihrer Struktur wurden Standardisierungsansätze des *IEEE Learning Technology Standards Committee*<sup>3</sup> und des *IMS Global Learning Consortium*<sup>4</sup> herangezogen. Die folgenden Vorschläge wurden dazu näher betrachtet:

#### IEEE Draft Standard for Learning Object Metadata (LOM), Version 6.1 (April 2001)

Erweiterung des Dublin Core Metadata Standards, Version 1.1 [Juli 1999]<sup>5</sup> – zur Angabe von Metadaten über Lehrobjekte jeder Art.

---

<sup>3</sup><http://ltsc.ieee.org>

<sup>4</sup><http://www.imsproject.org>

<sup>5</sup><http://www.dublincore.org/documents/1999/07/02/dces/>

Dieser Entwurf vereint und erweitert die Bemühungen des von der Europäischen Union geförderten Projekts „Ariadne“ zur Organisation von digitalem Lehrmaterial<sup>6</sup> und dem „IMS Learning Resource Meta-data Information Model“.

**IMS Question and Test Interoperability (QTI) Specification, Version 1.1 (März 2001)**  
Vorschlag zur Standardisierung von Aufgaben und deren Auswertungsanweisungen.

**IMS Learner Information Package Specification, Version 1.0 (März 2001)**  
Vorschlag zur Standardisierung von Daten über Lernende.

Da diese Standards jedoch noch im Fluss und zum Teil noch weit von einer offiziellen Verabschiedung durch ein entsprechendes Gremium entfernt sind, wurden lediglich einige „Design“-Entscheidungen übernommen; die XML-Daten lassen sich aber bei Bedarf mittels XSLT-Stylesheets verhältnismäßig einfach in die den Standards entsprechenden XML-Formate überführen.

## 8.2 Verwendete Frameworks

Während Avalon<sup>7</sup> ein schnittstellen-orientiertes Framework zur modularen Entwicklung von Java-Systemen zur Verfügung stellt, bietet Cocoon<sup>8</sup>, selbst auf Avalon basierend, ein umfangreiches Management zur Server-seitigen Verarbeitung von XML-Dateien. Diese beiden Frameworks werden im Folgenden kurz vorgestellt, um die Auswirkungen auf das System-Design von iTeach nachvollziehen zu können. Da beide Frameworks, insbesondere Cocoon, sehr umfangreich sind, sei an dieser Stelle für weitergehende Informationen auf die jeweiligen Homepages und Code-Dokumentationen verwiesen.

### 8.2.1 Apache Cocoon

Cocoon ist ein auf Java basierendes „Web Publishing Framework“, das im Januar 1999 als Projekt der ASF von Stefano Mazzocchi ins Leben gerufen wurde. Es ermöglicht die serverseitige Verarbeitung von XML-Dateien (die ihrerseits Logik enthalten können, das XML-File also dynamisch erzeugt werden kann) in beliebige Ausgabeformate (HTML, WML, PDF, ...). Nachdem sich abzeichnete, dass während der Entwicklung von Version 1 getroffene Design-Entscheidungen die Entwicklung größerer Projekte zu sehr behindern (was sich in erster Linie in schlechter Effizienz zeigte), wurde in Version 2 die Gesamtarchitektur neu entworfen, um diese Engpässe zu beseitigen.

Anfang Juni 2001 wurde die erste Betaversion von Cocoon2 veröffentlicht, die auch Grundlage für die Entwicklung von iTeach ist.

---

<sup>6</sup><http://www.ariadne-eu.org/>

<sup>7</sup><http://jakarta.apache.org/avalon>

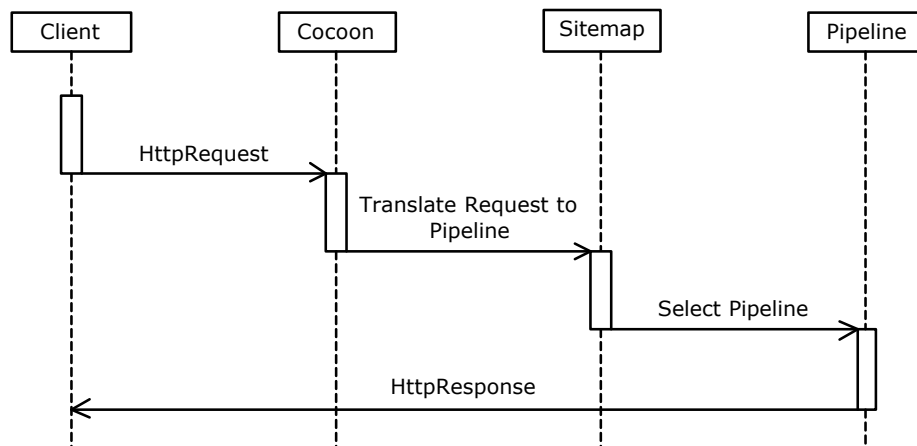
<sup>8</sup><http://xml.apache.org/cocoon2>

## Die Sitemap

Basis für Cocoon ist ein Servlet, das die Client-Requests beantwortet, indem eine Folge von Verarbeitungsschritten ausgeführt und die generierte Antwort an den Client zurückgeschickt wird.

Diese Folge von Verarbeitungsschritten wird durch ein Pipeline-Modell beschrieben, das aus mehreren Komponenten besteht<sup>9</sup>. Eine Pipeline besteht aus einem sog. „Generator“, der die Verarbeitung anhand des Client-Request-URI initiiert, null oder mehr sog. „Transformer“, die den Inhalt modifizieren und einem abschließenden „Serializer“, der die Antwort an den Client in einem beliebigen Format liefert. Die Komponenten einer Pipeline reichen die XML-Daten in Form von SAX-Events an die nächste Komponente weiter. Dies ist eine der wichtigsten Verbesserungen in Bezug auf die Effizienz gegenüber Version 1, bei der die Informationen in Form von DOM-Trees weitergereicht wurden.

Die Pipelines und die verfügbaren Komponenten werden in sog. „Sitemaps“ definiert; diese XML-Dateien stellen die zentralen Konfigurationsdateien von Cocoon dar und werden aus Effizienzgründen in Java class-Files übersetzt. Abbildung 8.1 zeigt den groben Ablauf eines Request-Response-Zyklus als UML-Sequenzdiagramm [4].



**Abbildung 8.1:** Verarbeitung eines Requests durch Cocoon 2 (nach der Cocoon2-Homepage).

Cocoon stellt eine Reihe von Standard-Komponenten zur Verfügung, die für viele Fälle bereits ausreichend sind, es können jedoch problemlos eigene Komponenten hinzugefügt werden, wovon im Rahmen von iTeach auch kräftig Gebrauch gemacht wurde (vgl. Abschnitt 8.3).

Neben diesen drei Typen von Komponenten existieren noch weitere, von denen die wichtigsten die folgenden sind:

<sup>9</sup>Der Begriff *Komponente* ist nicht ohne Grund gewählt; er taucht im Abschnitt über das Avalon-Framework wieder auf, vgl. Abschnitt 8.2.2).

**Matcher**

Dient der Auswahl einer Pipeline zu einem Request-URI.

**Action**

Dient der dynamischen Erzeugung von request-abhängigen Parametern, auf die in der Rest-Pipeline zugegriffen werden kann. Eine Action kann auch fehlschlagen, sodass auf diese Weise eine Pipeline aufgespaltet werden kann.

**Selector**

Erweiterung von Action in dem Sinne, dass eine Pipeline sich in mehr als zwei Äste aufspalten kann, z. B. stellt Cocoon Selectors für Client-Browser und Hosts zur Verfügung.

## 8.2.2 Apache Avalon

Was Avalon bietet, lässt sich am besten durch ein Zitat von der Avalon-Homepage<sup>10</sup> beschreiben:

“The Avalon project is an effort to create, design, develop and maintain a common framework and set of components for (server) applications written using the Java language. This framework is not a standalone product, but allows existing and yet to be created server applications to fit into a common platform and to share code, design and human resources.”

Die Design-Ideen und die dahinter stehenden Entwurfsmuster zu beschreiben, würde den Rahmen dieser Arbeit bei weitem sprengen, es sei also einmal mehr auf die Avalon-Homepage verwiesen, die eine gute Dokumentation des gesamten Projekts bereitstellt. Avalon besteht aus mehreren Teilprojekten, von denen in iTeach jedoch nicht alle eingesetzt werden; im Folgenden werden nun die verwendeten Teilprojekte mit Blick auf die Implementierung erläutert.

**Avalon Framework**

Auch für das eigentliche Avalon-Framework, eines der erwähnten Teilprojekte, sei mit einem Zitat begonnen:

“The Avalon framework consists of interfaces that define relationships between commonly used application components, best-of-practice pattern enforcements, and several lightweight convenience implementations of the generic components.

What that means is that we define the central interface `Component` as well as sub-interfaces like `Applications`. We also define the relationship (contract) a component has with peers, ancestors and children.”

---

<sup>10</sup><http://jakarta.apache.org/avalon>

Eine Komponente kann als eine „passive Klasse“ aufgefasst werden – sie wird von einer Fabrik instanziiert und je nach implementierten Interfaces werden in einer festen Reihenfolge verschiedene Methoden aufgerufen, um die Komponente zu initialisieren. Die Identifikation von Komponenten erfolgt über „Rollen“, der Begriff wurde dem Theater entnommen und verdeutlicht die Idee die hinter dieser „komponenten-basierten Programmierung“ steht: Komponenten entsprechen den Schauspielern und die Rolle wird vorwiegend durch die implementierten Interfaces festgelegt (das Drehbuch). In einer externen XML-Datei wird die Zuordnung von Komponenten zu Rollen festgelegt, sodass auf diese Weise sichergestellt wird, dass die einzelnen Komponenten problemlos austauschbar sind.

Existiert nur eine Komponente für eine Rolle (bei iTeach beispielsweise spielt die Klasse `DomainImpl` die Rolle `Domain`), wird der Klassenname aus der XML-Datei geladen und über die Fabrikmethode `lookup()` der Klasse `ComponentManager` initialisiert. Im Falle mehrerer möglicher Klassen (z. B. iTeachs verschiedene Testtypen), übernimmt die Klasse `ComponentSelector` die Initialisierung eines bestimmten Typs, der aufgrund eines „Nick names“ identifiziert wird.

### Avalon Excalibur

Das Teilprojekt Excalibur stellt u. a. eine Reihe von Default-Implementierungen zur Verfügung, die über die des Frameworks hinausgehen. Für nähere Informationen sei auf die Code-Dokumentation<sup>11</sup> verwiesen. iTeach verwendet einige dieser Klassen zur Vereinfachung des Komponentenzugriffs.

### LogKit

LogKit bietet komfortable Methoden zur Protokollierung von Programmabläufen; dabei kann der Logger auf vielfältige Weise konfiguriert werden. Neben mehreren „Log-Levels“ können unterschiedliche „Log-Targets“ definiert sowie verschiedene Logger hierarchisch organisiert werden.

iTeach verwendet Cocoons Logger, um die Ausgaben von Cocoon und iTeach zu bündeln, da in manchen iTeach-Komponenten automatisch auf Cocoons Logger zugegriffen wird.

## 8.3 Klassen und Pakete von iTeach

iTeach besteht aus ca. 65 Klassen und Interfaces, die auf 11 Pakete verteilt sind; im folgenden werden die wichtigsten Klassen vorgestellt, für eine ausführliche Beschreibung der einzelnen Klassen sei jedoch auf die Quellcode-Dokumentation verwiesen. Die Paket-Benennung erfolgte dabei in Anlehnung an die Paketstruktur von Cocoon.

---

<sup>11</sup><http://jakarta.apache.org/avalon/api/index.html>

Viele der Klassen und Interfaces in iTeach erben vom Component-Interface des Avalon Framework (entweder direkt oder indirekt), wodurch die jeweiligen Implementierungen durch Anpassung der Rollen-Definitionsdatei auf einfache Weise ausgetauscht werden können. Diese XML-Datei, `iteach.roles`, befindet sich analog zu Cocoon's Rollendatei im Basispaket `de.iteach`.

### 8.3.1 `de.iteach`

Dieses Paket enthält systemweite Klassen und Interfaces; neben den beiden Interfaces `Constants` und `Roles`, die zum einen allgemeine Konstanten wie die in den Metadaten enthaltenen Operator- und Aktionsnamen und zum anderen die „Nick names“ der Avalon-Komponenten in `iteach.roles` enthalten, finden sich hier die Klasse `CourseManager` und das Servlet `StartupServlet`.

`CourseManager` verwaltet in erster Linie allgemeine Konfigurationsparameter und die Avalon-Komponenten. Es ist als Singleton [22] implementiert und wird beim ersten Zugriff durch eine der Komponenten initialisiert. Die Initialisierung umfasst das Einlesen der Avalon-Rollen-Definitionen sowie der Konfigurationsdatei, deren Name als Initialisierungsparameter beim Laden von Cocoon dem Servlet `StartupServlet` übergeben wird.

### 8.3.2 `de.iteach.acting`

Dieses Paket enthält verschiedene Cocoon-Aktionen, die in der Sitemap definiert sind und beispielsweise für die Auswahl der anzuzeigenden Fragmente sowie Login und Registrierung verantwortlich sind. Jede Aktion liefert ein `Map`-Objekt zurück, auf dessen Inhalt in der restlichen Pipeline zugegriffen werden kann. Eine Aktion wird von Cocoon als fehlgeschlagen interpretiert, wenn der Rückgabewert `null` ist. Die einzelnen Aktionen von iTeach sind die folgenden:

#### **FileAuthenticatorAction**

Diese Aktion überprüft die Angaben im Login-Formular; existiert ein Eintrag zu angegebenem Login und Passwort in der Datei `users.xml`, werden die zugehörigen Benutzerdaten aus der entsprechenden XML-Datei geladen bzw. im Falle eines Gast-Logins ein neues `Student`-Objekt erzeugt. Abschließend wird eine neue Session erzeugt und das `Student`-Objekt in der Session gespeichert.

Die zurückgegebene `Map` enthält den Namen und das nächste Konzept des Benutzers.

#### **FragmentSelectorAction**

Mit dieser Aktion werden die Fragmente zu einem Konzept ausgewählt; dies entspricht dem Algorithmus 6.1 auf Seite 53. Als Parameter kann spezifiziert werden, ob Anzeigeregeln mit Aktion `HIDE` und/oder `SHRINK` ignoriert werden sollen. Abschließend wird das Benutzermodell aktualisiert (s. Abschnitt 6.4).

Die zurückgegebene Map enthält Namen und ID des Konzepts sowie die Liste der ausgewählten Fragmente mit der jeweiligen Aktion.

### **RegistrarAction**

Diese Aktion ist für die Verarbeitung von Neuanmeldungen verantwortlich. Nachdem überprüft wurde, ob alle eingegebenen Daten gültig sind (d. h. die zwei Passwörter müssen übereinstimmen, und die Login-Kennung darf noch nicht vergeben sein), wird ein entsprechendes Student-Objekt erzeugt und in einer neuen Session gespeichert. Zurückgegeben wird eine leere Map.

### **SessionInvalidatorAction**

Diese Aktion ist eine Erweiterung der von Cocoon zur Verfügung gestellten gleichnamigen Aktion; sie erklärt eine Session für ungültig, was einem regulären Logout entspricht. Dabei wird die Gesamtnutzungsdauer des Benutzers aktualisiert.

### **StereotypeInitAction**

Diese Aktion verarbeitet das Formular zur Stereotypen-Initialisierung des Benutzermodells. Es analysiert die Angaben des Benutzers und aktualisiert entsprechend den Angaben der Verarbeitungsanweisungen in einer separaten Datei das Benutzermodell (vgl. Anhang B und Abschnitt 6.4). Zurückgegeben wird das nächste dem Benutzer empfohlene Konzept.

### **TestHelpAction**

Diese Aktion ist für die Anzeige der aufgabenspezifischen Hilfe zuständig; dazu wird der Hilfe-Level bestimmt, indem überprüft wird, ob und wenn ja wann der letzte Zugriff auf diese Hilfe stattfand; die betroffenen Session-Attribute werden entsprechend aktualisiert. Zurückgegeben wird neben dem neuen Hilfelevel die XML-Datei mit den Testdaten und die Hauptkonzept-ID der Aufgabe.

### **TestResultAction**

Diese Aktion wird zur Auswertung einer Übungsaufgabe aufgerufen. Dazu wird als erstes – sofern der Benutzer in seinem Browser JavaScript-Unterstützung deaktiviert hat – die serverseitige, ansonsten die clientseitige Bearbeitungszeit bestimmt, die evtl. in der Session gespeicherten Hilfezugriffe gelöscht und die eigentliche Auswertung der Antwort an die betroffene Test-Klasse delegiert. Abschließend wird das Benutzermodell aktualisiert und neben dem Ergebnis (in %) die Inhaltsdatei der Aufgabe für die Generierung der Ergebnisseite zurückgegeben.

### **TestSelectorAction**

Analog zur `FragmentSelectorAction` wählt diese Aktion einen Test zum angeforderten Konzept aus.

### **UnregistratorAction**

Diese Aktion löscht alle Benutzerdaten; außer der XML-Datei mit den individuellen Benutzerdaten wird der entsprechende User in der Liste der angemeldeten

Benutzer gelöscht und die Session für ungültig erklärt. Zurückgegeben wird stets eine leere Map.

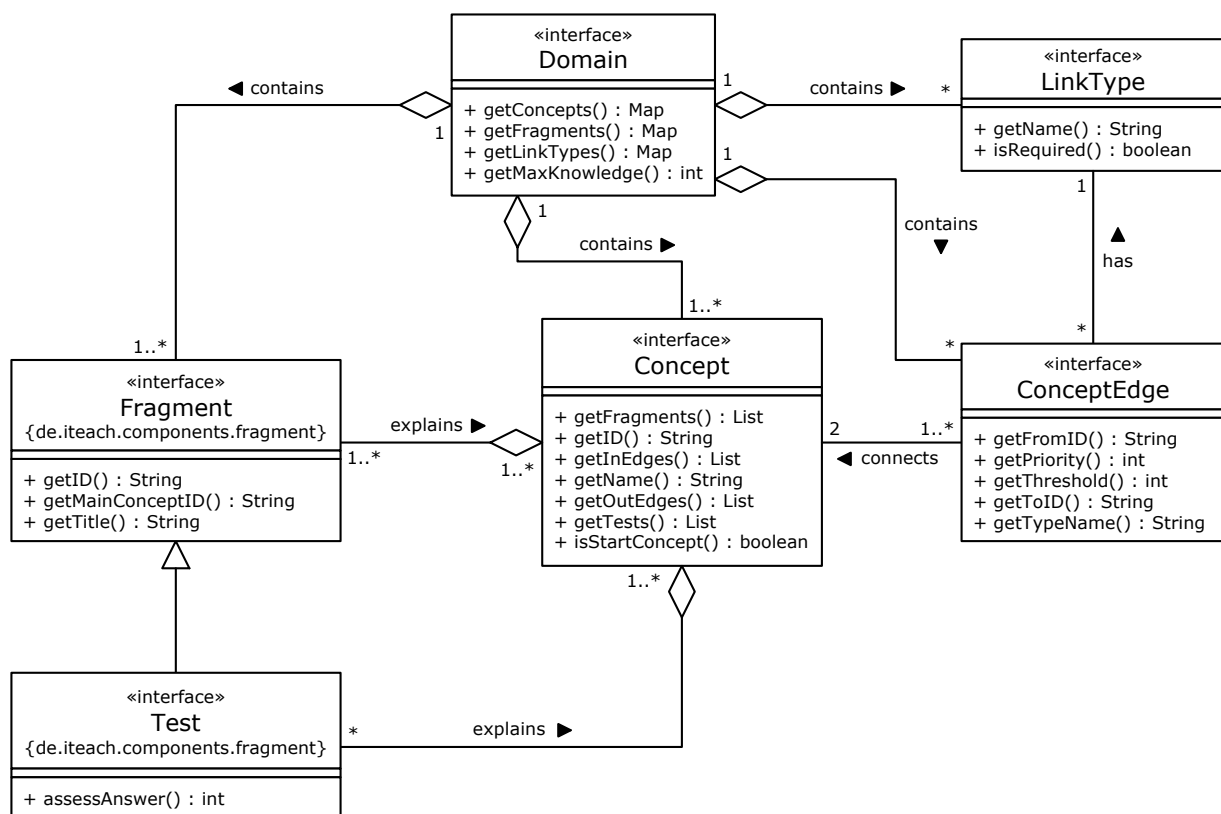
### UpdateUserModelAction

Diese Aktion schließlich verarbeitet manuelle Änderungen des Benutzermodells. Dazu werden die entsprechenden Werte gesetzt und das Benutzermodell mittels der Inferenzalgorithmen aktualisiert (s. Abschnitt 6.4).

### 8.3.3 de.iteach.components.domain

Dieses Paket enthält die vier Komponenten Domain, Concept, LinkType und ConceptEdge, jeweils als Interfaces mit einer Default-Implementierung. Diese Schnittstellen stellen analog zu Abschnitt 6.1.1 die Implementierungssicht auf die abstrakte Schicht des Domänenmodells dar.

Abbildung 8.2 zeigt die Interfaces des Domänenmodells und ihre gegenseitigen Abhängigkeiten.

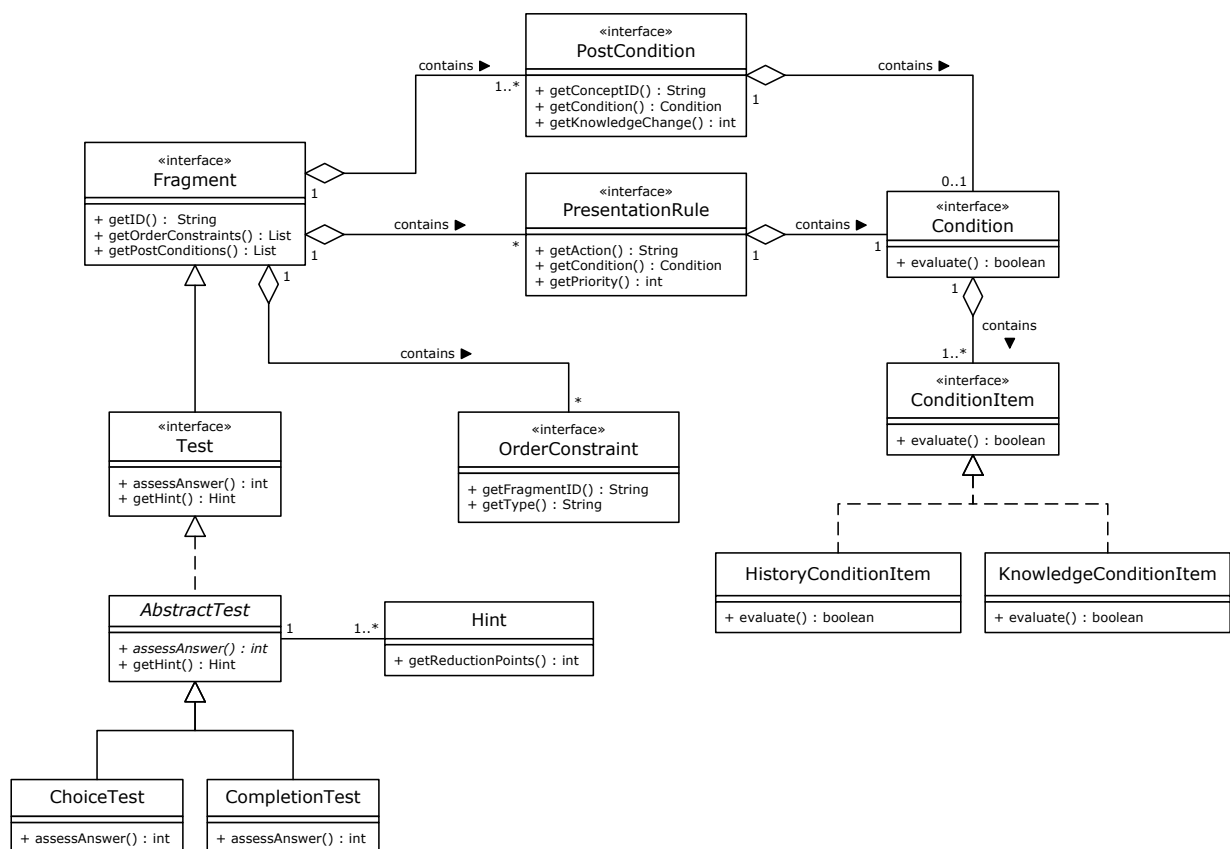


**Abbildung 8.2:** Die Klassen der abstrakten Schicht des Domänenmodells. Aus Gründen der Übersichtlichkeit wurde das von allen dargestellten Interfaces erweiterte Avalon-Interface Component weggelassen.

### 8.3.4 de.iteach.components.fragment[.metadata]

In diesen beiden Paketen befinden sich die Klassen und Interfaces für das Low-Level-Domänenmodell; neben den Interfaces `Fragment` und `Test` sind dies die verschiedenen Testtypen und Klassen für den Fragment- bzw. Testzugriff. Im Paket `metadata` befinden sich die Interfaces der komplexen Metadaten mit den jeweiligen Implementierungen.

Abbildung 8.3 zeigt diese Klassen und die gegenseitigen Abhängigkeiten.



**Abbildung 8.3:** Die Klassen der konkreten Schicht des Domänenmodells. Auch hier wurde auf das von allen Interfaces erweiterte Component-Interface verzichtet.

### 8.3.5 de.iteach.components.inference

Dieses Paket enthält die Schnittstelle `Inference` und die zugehörige Implementierung, die die Umsetzung der in Abschnitt 6.4 ab S. 55 beschriebenen Algorithmen enthält.

### 8.3.6 `de.iteach.components.student`

In diesem Paket befinden sich neben dem `Student`-Interface die entsprechende Implementierung, die Klasse `User-Manager`, die für das Laden, Speichern und Initialisieren von `Student`-Objekten zuständig ist, sowie die `PageFull`-Heuristik mit ihrer Default-Implementierung.

### 8.3.7 `de.iteach.generation`

Dieses Paket enthält einzig die Klasse `ModelGenerator`, die als Grundlage für die Benutzermodellansicht dient. Das Benutzermodell (und weitere hilfreiche Daten zur Ansicht wie die durchschnittliche Bearbeitungszeit von Übungsaufgaben u. ä.) wird als DOM-Tree [2] erzeugt und von XSLT-Stylesheets weiterverarbeitet.

### 8.3.8 `de.iteach.tests`

Dieses Paket enthält die verschiedenen JUnit Tests.

### 8.3.9 `de.iteach.transformation`

In diesem Paket befinden sich die einzelnen Transformer von iTeach, die in unterschiedlichen Pipelines aufgerufen werden, um den Inhalt zu modifizieren. Folgende Transformer sind verfügbar:

#### **FragmentFilterTransformer**

Dieser Transformer benötigt als Konfigurationsparameter den Namen und Namespace-URI eines Attributs zur Erkennung der Fragmente im SAX-Stream. Ferner erhält er als Ergebnis der zuvor erfolgreich aufgerufenen Aktion `FragmentSelectorAction` eine Liste der anzuzeigenden Fragmente; nur die Fragmente mit den in dieser Liste auftauchenden IDs werden an die nächste Komponente in der Pipeline weitergereicht. Hierbei werden natürlich die in der Liste ebenfalls enthaltenen Fragment-Aktionen berücksichtigt, sodass letztlich nur `SHOW`-Fragmente angezeigt werden; für `SHRINK`-Fragmente wird hingegen ein Link zu diesem Fragment mit dem Fragmenttitel als Verweistext erzeugt.

Aus Effizienzgründen sollte dieser Transformer möglichst weit vorn (am besten als erster Transformer) in der Pipeline ausgeführt werden.

#### **GlossaryTransformer**

Der `GlossaryTransformer` erhält zur besseren Übersicht als Parameter einen URL zu einer separaten Konfigurationsdatei. Über diese Datei kann das Verhalten der automatischen Erkennung und Verlinkung von Glossareinträgen gesteuert werden; die Datei enthält ausführliche Beschreibungen der einzelnen Parameter (vgl. Anhang B).

### LinkTransformer

Der `LinkTransformer` erzeugt am Ende einer Seite eine Liste mit Verweisen zum nächsten empfohlenen Konzept, zum Vor- und Zurückblättern in der Browsing-History sowie zu allen Folgekonzepten (vgl. Abschnitt 7.1.1 auf Seite 62).

### LinkAnnotator

Der `LinkAnnotator` ist für die adaptive Annotation von Verweisen zuständig; er wird wie der `GlossaryTransformer` über eine separate Konfigurationsdatei gesteuert, in der die einzelnen Parameter ausführlich erläutert sind. Die eigentliche Annotation (also beispielsweise das Hinzufügen von Piktogrammen) findet erst später im Pipeline-Prozess statt; der `LinkAnnotator` fügt lediglich je nach Art des Links ein Attribut `class` mit einem der Werte `not-ready`, `ready`, `partly-known`, `known`, `current` oder `next` (für Konzepte) bzw. `not-read` oder `read` für Fragmente hinzu.

Durch die Trennung vom eigentlichen Layout, das in der Regel von einem XSLT-Stylesheet übernommen wird, können auch die adaptiven Komponenten in ihrer Darstellung beeinflusst (oder durch ein entsprechendes Stylesheet ganz ignoriert) werden.

### 8.3.10 `de.iteach.utils`

Enthält systemweite Hilfsfunktionen zur Behandlung von Strings (`StringUtil`) und XML-Dateien (`XMLUtil`). Alle Methoden dieser beiden Klassen sind `static`.

Die wichtigsten Verzeichnisse und Dateien werden in Anhang B, Hinweise zur Installation und Konfiguration in Anhang C beschrieben.

# Kapitel 9

## Ausblick

*„Der Mensch ist immer noch der beste Computer.“*

*John F. Kennedy*

Im Rahmen dieser Arbeit wurde ein adaptives hypermediales Lehrsystem für den Einsatz im WWW vorgestellt, bei dessen Entwicklung besonders die Aspekte der Wiederverwendbarkeit und Erweiterbarkeit im Vordergrund standen. Erreicht wurde dies durch die vollständige Implementierung in Java, wodurch das System unabhängig von der verwendeten Betriebssystem-Plattform ist und durch die ausschließliche Datenhaltung in XML, mit der auf den Einsatz von Datenbanken verzichtet wurde. Die Verarbeitung erfolgt vollständig serverseitig, womit sich die Anforderungen an die Client-Software auf einen (frei verfügbaren) Webbrowser beschränken. Als zugrunde liegende Plattform wurde Apaches Cocoon-Framework gewählt, um die Verarbeitung der Requests in einzelne Schritte zerlegen zu können und diese einzelnen Komponenten leicht austauschbar zu halten. Da sie alle mithilfe von Fabrikmethoden erzeugt werden und nur über Schnittstellen angesprochen werden, sind andere Codeteile nicht betroffen, solange die entsprechenden Schnittstellen implementiert werden.

Im Zuge der rasanten Verbreitung von XML als universelles Datenformat im WWW lag die Idee nahe, ein adaptives Lehrsystem zu entwickeln, dessen Daten *vollständig* in XML gehalten werden, um den einfachen Datenaustausch zwischen Anwendungen zu ermöglichen und zu fördern. Da die in Abschnitt 8.1.2 beschriebenen Standardisierungsbemühungen noch nicht abgeschlossen sind, wurden lediglich die hinter den Standards stehenden Design-Überlegungen übernommen, insbesondere für die Metadaten der einzelnen Fragmente aber letztlich eine eigene Formulierung verwendet. Mithilfe von XSLT Stylesheets lassen sich aber die Formate bei Bedarf ohne größeren Aufwand in die entsprechenden XML-Strukturen transformieren, um so den Austausch von Lehrinhalten und bei Bedarf auch von Benutzerdaten zwischen verschiedenen Anwendungen zu ermöglichen.

Die in Abschnitt 4.5 beschriebenen adaptiven Lehrsysteme basieren zum größten Teil auf proprietären Formaten, vornehmlich HTML, womit neben dem Datenaustausch vor allem die Änderung des Layouts und die Ausgabe auf anderen Medien wie das Portable

Document Format (PDF) von Adobe sich als schwierig gestalten dürften. Für iTeach bzw. Cocoon stellt dies kein Problem dar, da lediglich die entsprechenden Stylesheets geschrieben werden müssen, um die entsprechende Ausgabe aus *demselden* Datenfile zu erzeugen.

Auf der Grundlage verschiedener bestehender Kriterienkataloge mehrerer Fachbereiche wurde ein Katalog zur Evaluation von Online-Kursen entwickelt, der eine umfassende Beurteilung von Kursen im WWW und ähnlicher Lehrmaterialien ermöglicht. Mithilfe dieses Katalogs kann die Qualität anderer Lehrinhalte beurteilt werden und – natürlich unter Berücksichtigung der Urheberrechte – in iTeach integriert werden (vgl. dazu auch Abschnitt 9.1.1).

Um die Funktionsweise des Systems nachvollziehen zu können, wurde ein kleiner Teil der vorliegenden Ausarbeitung als Beispielinhalt eingesetzt; dieser Kurs „Einführung in Adaptive Hypermedia“ umfasst in erster Linie die Kapitel 3 und 4 und wurde um einfache Übungsaufgaben und Tests erweitert.

Das in iTeach implementierte Domänen- und Benutzermodell folgt dem Entwurf klassischer Ansätze, die sich in der Vergangenheit bereits als sinnvoll und praxistauglich erwiesen haben. Auch die adaptiven Techniken basieren letztlich auf der Analyse der in anderen bestehenden adaptiven Lehrsystemen implementierten Methoden und Techniken, wobei darauf geachtet wurde, den Benutzer während des Lernens nicht zu sehr einzuschränken.

Frei nach der Äußerung Kennedys zu Beginn dieses Kapitels folgt iTeach dem Ansatz „Suggestions instead of restrictions“; Annahmen über das Wissen des individuellen Benutzers können von diesem überschrieben und damit rückgängig gemacht werden. Obwohl dieser Ansatz den Ergebnissen von Spechts Untersuchungen [47] zur Steigerung der Lerneffizienz durch Link-Annotationen und inkrementelle Links widerspricht (vgl. Abschnitt 3.7), scheint es aus eigener Erfahrung sinnvoller zu sein, den Benutzer hilfreich zu unterstützen und ihn zu entlasten, als ihn in „vorgefertigte Bahnen“ zu zwingen.

Zusätzlich verfügt iTeach über ein Glossar, dessen Einträge sowohl explizit im Inhalt verlinkt als auch automatisch vom System im Text erkannt werden können. Um den Textfluss nicht zu stören und das gesamte Erscheinungsbild ansprechend zu halten, ist der Erkennungsalgorithmus tolerant auf Wortendungen; über einen Konfigurationsparameter kann gesteuert werden, ob und wie viele Zeichen am Ende eines Wortes stehen dürfen, um es als gültigen Glossareintrag zu erkennen und zu verlinken. Auf diese Weise lassen sich Phänomene wie Genitiv-s und Pluralformen behandeln und flüssig als das ganze Wort umfassenden Hyperlink darstellen.

Beim Design der Benutzeroberfläche wurde darauf geachtet, dass die Link-Annotationen leicht erkennbar sind und ihr Aussehen konsistent ist. Anhand von Textbeschreibungen der Piktogramme (in den gängigen Webbrowsern als sog. „Tooltips“ zu erkennen) steht zusätzlich eine klare textuelle Beschreibung zur Verfügung. Ferner enthält die

auf jeder Seite enthaltene Kopfleiste Buttons für die häufigsten Funktionen; auch beim Entwurf dieser Buttons wurde auf einheitliches Aussehen geachtet. Aufgrund der Realisierung über XML und XSLT lässt sich das *gesamte* Layout ändern, ohne Änderungen im eigentlichen Quellcode vornehmen zu müssen.

Auch wenn diese vorgestellten Funktionen wichtige Voraussetzungen für einen erfolgreichen Einsatz in der Praxis darstellen, gibt es verschiedene Bereiche des Systems, die weiter ausgebaut, erweitert und verbessert werden können; Anregungen dazu werden nun beschrieben.

## 9.1 Erweiterungsmöglichkeiten

Die in diesem Abschnitt erläuterten Erweiterungen des vorgestellten Systems verteilen sich auf die drei Bereiche *Autoren-Unterstützung*, *Lerner-Unterstützung* und *technische Erweiterungen*. Die folgenden Aufstellungen bieten eine Reihe von Anregungen, wie das beschriebene System weiter ausgebaut werden kann.

### 9.1.1 Autoren-Unterstützung

Die nahe liegendste Erweiterung von iTeach ist ein Autorensystem, das die Integration vorhandener Kursmaterialien sowie die Entwicklung neuer Kurse für iTeach und insbesondere die Eingabe der Fragment-Metadaten erleichtert.

#### Grafischer Ontologie-Editor

Ein Autorensystem mit grafischer Oberfläche, das ein komfortables Frontend zur Eingabe des Konzeptgraphen (idealerweise direkt als Graph) und der an den einzelnen Konzeptknoten „hängenden“ Fragmente bietet. Neben Inhalt und gegenseitigen Abhängigkeiten ist die einfache Eingabe von Präsentationsregeln der einzelnen Fragmente erforderlich, evtl. mit intelligenter Unterstützung in Form von Regel-Assistenten, die eine von sich aus sinnvolle Präsentation ermöglichen.

#### Offene Architektur

Als Erweiterung des Ontologie-Editors sollten externe Quellen im WWW integriert werden können; entweder indem diese direkt in den Kurs importiert werden (besonders bei Texten sinnvoll), oder entsprechende Links zu den Adressen erzeugt werden.

### 9.1.2 Lerner-Unterstützung

Neben der Autorenunterstützung lässt sich natürlich die Benutzer-orientierte Unterstützung weiter ausbauen.

#### Weitere Techniken

Auf der Basis von empirischen Untersuchungen können weitere adaptive Techniken wie Fragment-Sortierung, Fragment-Varianten und das Ausblenden von

Links implementiert werden. Dabei sollte vornehmlich die Akzeptanz und die erbrachte Steigerung der Lerneffizienz durch die Benutzer berücksichtigt werden.

### **Eigene Lernziele und Trails**

Anstatt das aktuelle lokale Lernziel vollständig vom System bestimmen zu lassen, könnten sowohl Autoren als auch Benutzer die Möglichkeit erhalten, Lernziele und Folgen von Konzepten zu definieren, vergleichbar den „Trails“ im Java Tutorial von Sun<sup>1</sup>.

### **Dynamische Änderung der Adaptionstechniken**

Je nach Interessenlage und Wissensstand des Benutzers könnten sich im Laufe der Nutzung die Adaptionstechniken und ihre Arbeitsweise anpassen; beispielsweise ließe sich auf diese Weise eine anfangs stärkere Führung des Lerners durch Ausblenden bzw. Deaktivieren von Verweisen erreichen, die nach und nach weniger restriktiv wird. Die damit einhergehenden kognitiven Anforderungen sollten jedoch in angemessenem Umfang analysiert werden, um eine Verbesserung der Lernausbeute zu erreichen. Auch unter diesem Aspekt könnten verschiedene empirische Untersuchungen zur Lerneffizienz anhand von adaptiven Techniken durchgeführt werden.

### **Kollaboratives Lernen und Problemlösung**

Ein zunehmend wichtiger Aspekt ist die Unterstützung der Kommunikation von Lernenden im Allgemeinen (beispielsweise durch Chats und Diskussionsforen) und die gemeinsame Problemlösung im Speziellen. Verschiedene Arbeiten auch im Bereich adaptiver Lehrsysteme im WWW gehen bereits in diese Richtung (vgl. dazu auch Abschnitt 2.3.2); generell stellen diese die Brücke zwischen adaptiven Einzel-Lerner-Systemen und kollaborativen Tutorsystemen dar.

### **Gestaltung der Oberfläche nach pädagogischen Gesichtspunkten**

Auch wenn bei der Entwicklung auf eine konsistente und ansprechende Benutzerschnittstelle geachtet wurde, kann sie unter Berücksichtigung von pädagogischen und lernpsychologischen Aspekten weiter verbessert werden. Neben der Oberfläche an sich lassen sich insbesondere auch das Feedback des Systems dem Lernenden gegenüber intelligenter gestalten und gezielt bekannte Schwachstellen wie Motivationsprobleme durch abwechslungsreiche Interaktivität und entsprechende Rückmeldungen des Systems bekämpfen.

### **Interessenprofile**

Anhand von Interessenprofilen kann der einzelne Benutzer Angaben zu seinen Wünschen und Vorlieben machen, beispielsweise die bevorzugten Medientypen, wie in [47] beschrieben; ferner sollte er die Möglichkeit haben, diese zu einem späteren Zeitpunkt wieder ändern zu können.

---

<sup>1</sup><http://java.sun.com/docs/books/tutorial/index.html>

### 9.1.3 Technische Erweiterungen

Auch eine Reihe von kleineren Änderungen, die die Implementierung und allgemein die technische Seite betreffen, könnten sich in Zukunft und bei längerem Praxiseinsatz als sinnvoll und hilfreich erweisen. Die folgende Aufstellung enthält dazu erste Anregungen.

#### **Effizienz**

Besonders bei vielen gleichzeitigen Zugriffen könnte ein intelligenteres Update der Benutzerdaten erforderlich werden um die Serverlast zu verringern. Prinzipiell ist aber bei jedem Zugriff eine Aktualisierung des Studentenmodells erforderlich, und dieses sollte auch sofort persistent gemacht werden um lästigen Datenverlust zu verhindern – dementsprechend besteht die einzige effektive Alternative darin, Teile der Daten clientseitig zu verarbeiten, was aber wiederum entsprechende Anforderungen an den Client und seine Ausstattung stellt.

#### **Verbesserte Aufgabenauswertung**

Die Auswertung von Übungsaufgaben, insbesondere von Lückentext, ist in iTeach nur anhand von einfachen Stringvergleichen implementiert. Durch Verwendung von beispielsweise regulären Ausdrücken können die gegebenen Antworten besser beurteilt werden.

#### **Datenschutz**

Aufgrund der teilweise persönlichen Daten der Benutzer, ist eine verschlüsselte Speicherung bzw. die Sperrung der Daten für Dritte erforderlich. Anstatt der in iTeach erforderlichen Zugriffssperre der Verzeichnisse auf Webserver-Ebene kann ein eigener Schauspielhaus die Sicherheit weiter erhöhen.

#### **Link-Vorschau und Multi-Links**

Wie in [32] beschrieben, kann u. U. eine Inhaltsangabe einer Seite und die Zusammenfassung von Links zu einem einzigen wünschenswert sein sowie Lesbarkeit und Textfluss verbessern.

#### **Bessere Heuristiken**

Neben einer Client-anhängigen Page-Full-Heuristik, die z. B. auf der Bildschirmauflösung des Client-Rechners basiert, können die in den Inferenz-Algorithmen enthaltenen Heuristiken verändert werden und auf Basis empirischer Lerneffizienz-Untersuchungen sowie Benutzer-Befragenden optimiert werden.

## 9.2 Abschließender Ausblick

Zum Schluss soll noch ein kurzer Blick in die Zukunft der computerbasierten Ausbildung im WWW geworfen werden. Blumstengel nennt drei Faktoren, die für eine vermehrte Nutzung computer- bzw. webbasierter Lernsoftware im universitären Rahmen erfüllt werden müssen [3]:

- Verbesserung der Zusammengeschweißten
- Verringerung der Kosten
- Erhöhung der Qualität

Diese Faktoren finden durch die vermehrte – auch politisch und damit finanziell unterstützte – Einrichtung von virtuellen Hochschulen (vgl. Abschnitt 2.3.2) gegenwärtig relativ große Beachtung; neben dem Kostenfragen ist vor allem der letzte Faktor entscheidend für die Zukunft der computergestützten Ausbildung. Für eine andauernd hohe Qualität bei der Entwicklung von Online-Lehrmaterial ist die fachübergreifende Zusammenarbeit von Pädagogen, Psychologen, Informatikern und Experten der vermittelten Fachrichtungen erforderlich, um Motivations- und Akzeptanzprobleme auf Seite der Lernenden zu minimieren sowie die Lerneffektivität und -effizienz zu maximieren.

Trotz der zunehmenden Verfügbarkeit von Lehrsystemen im WWW bleibt abzuwarten, inwieweit diese die klassische Ausbildung verdrängen werden, wie von Perelman vorausgesagt wurde (s. S. 6). Bei der zukünftigen Entwicklung solcher Systeme sollte aus den in Kapitel 2 geschilderten Gründen die gegenseitige Ergänzung klassischer und computerbasierter Ausbildungsformen im Vordergrund stehen. Beaumont bringt die Vorteile einer solchen Kombination auf den Punkt [1]:

“The advantage of computer aided instruction is not its ability to water down or liven up its instruction to the level of a bored or dejected student, but in its potential for reducing the teacher’s routine workload, leaving him more time to devote to individual students’ problems, and in its availability at those times when the student has a desire to learn, whatever time that may be.”

Schulmeister zitiert in diesem Zusammenhang Stebler et al. [44]:

„Wir müssen Lerngelegenheiten schaffen, die das Vorwissen der Schüler aufgreifen, der Situationsbezogenheit des Denkens Rechnung tragen und Lernen als selbstgesteuerten Wissensaufbau im Rahmen von Lern- oder Forschungsgemeinschaften konzipieren. Wir brauchen interaktive Lehr-Lern-Umgebungen.“

Dies spricht einen weiteren, in der Zukunft zunehmend wichtiger werdenden Aspekt an: Die Individualisierung der Lernsoftware; hier sind in erster Linie umfassende empirische Untersuchungen erforderlich, um die bereits durchgeführten Experimente und deren Ergebnisse zu stützen und ferner um die Forschung auf effektive Lernerunterstützung zu konzentrieren.

Dabei erscheint die Möglichkeit eines Systems, sich an den individuellen Benutzer anzupassen, auch unabhängig von Belegen durch statistisch fundierte Experimente,

äußerst viel versprechend – gerade im Hinblick auf die Integration externer Ressourcen im WWW und die damit verbundene Informationsflut. Als solide Basis für eine solche Integration scheinen die in Abschnitt 8.1.2 genannten Standardisierungsbemühungen und ihre Umsetzung in XML als universelles Datenformat sehr gut geeignet.

# Anhang A

## Evaluation der Java-Kurse

Die folgenden Tabellen enthalten die einzelnen Beurteilungen der Java-Kurse (vgl. Kapitel 5 ab Seite 37). Zur leichteren Zuordnung sind hinter den Kursnummern die Erstautoren angegeben.

### A.1 Allgemeines

Kriterium	1 (SUN)	2 (ECK)	3 (ECKEL)	4 (GOSLING)	5 (KJELL)
<b>Erforderliche Vorkenntnisse</b>	Keine Kenntnisse	Keine Kenntnisse	Grundlagen	C/C++ Kenntnisse	Keine Kenntnisse
<b>Angemessene Präsentation</b>	Ja	Zu langweilig	Ja	Nein	Ja
<b>Umfang des Kurses</b>	Sehr umfangreich	Umfangreich	Sehr umfangreich	Oberflächlich	Oberflächlich
<b>Sprache</b>	Englisch	Englisch	Englisch	Englisch	Englisch

Kriterium	6 (KRÜGER)	7 (LEMAY)	8 (MUKHI)	9 (PAWLAN)	10 (SCHMIDT)
<b>Erforderliche Vorkenntnisse</b>	Grundlagen	Keine Kenntnisse	Grundlagen	Grundlagen	Keine Kenntnisse
<b>Angemessene Präsentation</b>	Ja	Ja	Nein	Ja	Zu langweilig
<b>Umfang des Kurses</b>	Sehr umfangreich	Umfangreich	Oberflächlich	Umfangreich	Umfangreich
<b>Sprache</b>	Deutsch	Deutsch	Englisch	Englisch	Englisch

## A.2 Inhalt

Kriterium	1 (SUN)	2 (ECK)	3 (ECKEL)	4 (GOSLING)	5 (KJELL)
Vermittlung von Praxis/Theorie	Ja/Wenig	Ja/Ja	Ja/Ja	Ja/Wenig	Ja/Ja
Korrekt und umfassend	Ja	Ja	Ja	Zu kurz	Zu kurz
JDK-Version angegeben	Ja	Ja	Ja	Nein (veraltet)	Ja
Texte knapp und präzise	Ja	Ja	Ja	Ja	Ja
(Code-)Beispiele	Ja	Ja	Ja	Ja	Ja
Beispiele sind aussagekräftig	Ja	Ja	Ja	Ja	Ja
Fragen/Programmieraufgaben	Ja/Ja (nicht immer)	Ja/Ja	Nein/Ja	Nein/Nein	Ja/Ja (und Quiz)
Übungen sind motivierend	Ja	Ja	Ja	—	Ja
Lösungen vorhanden	Ja	Ja	Nein	—	Ja
Schwierigkeitsgrade angegeben	Nein	Nein	Ja	—	Nein
Kapitel-Zusammenfassung	Meistens	Nein	Ja	Meistens	Nein

Kriterium	6 (KRÜGER)	7 (LEMAY)	8 (MUKHI)	9 (PAWLAN)	10 (SCHMIDT)
Vermittlung von Praxis/Theorie	Ja/Ja	Ja/Wenig	Ja/Wenig	Ja/Wenig	Ja
Korrekt und umfassend	Ja	Manchmal zu kurz	Zu kurz	Ja	Ja
JDK-Version angegeben	Ja	Ja	Nein	Ja	Nein
Texte knapp und präzise	Ja	Ja	Nicht präzise	Ja	Ja
(Code-)Beispiele	Ja	Ja	Ja	Ja	Ja
Beispiele sind aussagekräftig	Ja	Ja	Ja	Ja	Ja
Fragen/Programmieraufgaben	Nein/Nein	Ja/Nein (nur Fragen)	Nein/Nein	Nein/Nein	Ja/Ja
Übungen sind motivierend	—	Nein	—	—	Ja
Lösungen vorhanden	—	Ja	—	—	Ja
Schwierigkeitsgrade angegeben	—	Nein	—	—	Nein
Kapitel-Zusammenfassung	Ja (Kurz)	Ja	Nein	Ja	Ja

## A.3 Präsentation

Kriterium	1 (SUN)	2 (ECK)	3 (ECKEL)	4 (GOSLING)	5 (KJELL)
Einzelne Kapitel	Ja	Ja	Ja	Ja	Ja
Kapitel sind überschaubar	Ja	Ja	Ja	Ja	Ja
Lernziele angegeben	Ja	Ja	Ja	Ja	Ja
Andere JDK-Versionen hervorgehoben	Ja	Ja	Ja	Nein	Ja
Motivierender Schreibstil	Ja	Etwas trocken	Ja	Ja	Ja
Hervorhebungen im Text	Nein	Wenig	Nein	Nein	Nein
Kursinterne Links	Ja	Ja	Nein (nur Fußnoten)	Nein	Nein
Externe Links	Ja	Nein	Nein	Nein	Nein
Überlegter Einsatz der Links	Ja	Ja	—	—	—
Kennzeichnung der Linkarten (intern/extern)	Ja	Nein	—	—	—
Grafiken und Diagramme	Ja	Wenig	Ja	Ja	Ja
Andere Multimedia-Elemente	Nein	Nein	Nein	Nein	Nein
Ansprechendes und übersichtliches Layout	Ja	Nein	Mäßig	Nein	Nein
Syntax-Highlighting bei Code Snippets	Nein	Nein	Ja	Nein	Nein
Hervorhebung von Beispielen etc.	Ja (<hr>)	Nein	Nein	Nein	Nein
Hervorhebung ist konsistent	Ja	—	—	—	—
Tippfehler/HTML-Fehler	Wenig/Nein	Wenig/Nein	Wenig/Ja	Wenig/Ja	Wenig/Nein

Kriterium	6 (KRÜGER)	7 (LEMAY)	8 (MUKHI)	9 (PAWLAN)	10 (SCHMIDT)
Einzelne Kapitel	Ja	Ja	Ja	Ja	Ja
Kapitel sind überschaubar	Ja	Ja	Nein	Ja	Ja
Lernziele angegeben	Ja	Ja	Nein	Ja	Ja
Andere JDK-Versionen hervorgehoben	Ja	Ja	Nein	Nein	Nein
Motivierender Schreibstil	Etwas trocken	Etwas albern	Albern	Ja	Ja
Hervorhebungen im Text	Wenig	Nein	Nein	Nein	Nein
Kursinterne Links	Ja	Nein	Nein	Ja	Nein
Externe Links	Ja	Nein	Nein	Nein	Nein
Überlegter Einsatz der Links	Ja	—	—	Ja	—
Kennzeichnung der Linkarten (intern/extern)	Nein	—	—	Nein	—
Grafiken und Diagramme	Ja	Wenig	Nein	Ja	Ja
Andere Multimedia-Elemente	Nein	Nein	Nein	Nein	Nein
Ansprechendes und übersichtliches Layout	Ja	Nein	Nein	Nein	Nein
Syntax-Highlighting bei Code Snippets	Ja	Nein	Nein	Nein	Nein
Hervorhebung von Beispielen etc.	Ja (Rand, Hintergrund)	Ja (Icons)	Nein	Nein	Nein
Hervorhebung ist konsistent	Ja	Ja	—	—	—
Tippfehler/HTML-Fehler	Wenig/Ja	Wenig/Nein	Wenig/Nein	Wenig/Nein	Wenig/Nein

## A.4 Bedienung

Kriterium	1 (SUN)	2 (ECK)	3 (ECKEL)	4 (GOSLING)	5 (KJELL)
Klare Führung	Ja	Ja	Ja	Ja	Ja
Leichte Orientierung	Ja (im Trail)	Ja	Ja	Ja	Nein
Kleine Dateien	Ja	Ja	Nein	Ja	Ja
Komprimierung der Multimedia-Daten	Ja	Ja	Ja	Ja	Ja
Glossar vorhanden	Ja	Nein	Nein	Nein	Nein
Glossar-Einträge verlinkt	Nein	—	—	—	—
Stichwortindex vorhanden	Nein	Nein	Ja	Nein	Ja
Index-Einträge verlinkt	—	—	Ja	—	Ja
Volltextsuche vorhanden	Ja (online)	Nein	Nein	Nein	Nein

Kriterium	6 (KRÜGER)	7 (LEMAY)	8 (MUKHI)	9 (PAWLAN)	10 (SCHMIDT)
Klare Führung	Ja	Ja	Ja	Ja	Ja
Leichte Orientierung	Ja	Ja	Nein	Ja	Mäßig
Kleine Dateien	Ja	Nein	Ja	Ja	Nein
Komprimierung der Multimedia-Daten	Ja	Ja	Ja	Ja	Ja
Glossar vorhanden	Nein	Nein	Nein	Nein	Ja (kapitelweise)
Glossar-Einträge verlinkt	—	—	—	—	Nein
Stichwortindex vorhanden	Ja	Ja	Nein	Nein	Ja
Index-Einträge verlinkt	Ja	Ja	—	—	Nein
Volltextsuche vorhanden	Ja	Nein	Nein	Nein	Nein

## A.5 Gesamtbeurteilung

Die Gesamtbeurteilung in den Bereichen Inhalt, Layout und erforderliche Änderungen gründet auf den vorhergehenden Kriterien und ist natürlich mit einer gewissen Subjektivität belastet. Im folgenden werden aus diesem Grund kurze Begründungen der erteilten Noten gegeben.

Zur Bildung der Gesamtnote wurden die drei Einzelnoten gleichmäßig gewichtet; die Zahlen sind klassische Schulnoten, feinere Abstufungen werden durch + und – angegeben.

### Sun Microsystems: „The Java Tutorial“ [Nr. 1]

**Inhalt:** Sehr umfassend, aber weniger Hintergründe als bei Eckel. Keine Übungen (1).

**Layout:** Mehr farbliche Hervorhebungen wären schön, aussagekräftigere Piktogramme zur Link-Annotation (1–).

**Änderungen:** Fehlende Übungen und Zusammenfassungen hinzufügen (1).

### D.J. Eck: „Introduction to Programming using Java“ [Nr. 2]

**Inhalt:** Nicht so umfassend wie das Tutorial von Sun oder Eckels „Thinking in Java“ (2).

**Layout:** Praktisch nur Text und Code-Beispiele, wirkt ziemlich langweilig (3).

**Änderungen:** Besseres Layout (3).

### B. Eckel: „Thinking in Java“ [Nr. 3]

**Inhalt:** Sehr umfassend und detailliert (1+).

**Layout:** Zu wenig Hervorhebungen, praktisch keine Hyperlinks, selbst URLs im Text sind nicht verlinkt (2).

**Änderungen:** Kapitel kleiner, besseres Layout (1–).

### J. Gosling, H. McGilton: „The Java Language Environment – A White Paper“ [Nr. 4]

**Inhalt:** Zu oberflächlich, veraltet, bezieht sich mehr auf Java allgemein (3).

**Layout:** Nur Text und Code-Beispiele, wirkt zu langweilig (3).

**Änderungen:** Ausführlicherer Text, Übungen und besseres Layout (4).

### B. Kjell: „Introduction to Computer Science using Java“ [Nr. 5]

**Inhalt:** Zu oberflächlich. Erklärung der Konzepte praktisch nur über Code-Beispiele. Sehr interaktiv (je Konzept eine Frage!), Quiz und „Flash Cards“ sind motivierend (3).

**Layout:** Aufmachung ist nicht schön, Orientierung innerhalb des Kurses nur über Back-Button (3).

**Änderungen:** Ausführlicherer Text, Übungen und Quiz-Fragen könnten übernommen werden (3–).

**G. Krüger: „Goto Java2“ [Nr. 6]**

**Inhalt:** Sehr umfassend und gut erklärt, manchmal etwas trocken. Keinerlei Interaktivität (2+).

**Layout:** Guter Einsatz von Markierungen (Tipps, Warnungen etc.) und Hintergrundfarben (1).

**Änderungen:** Kapitel kleiner, Übungen (1-).

**L. Lemay: „Java in 21 Tagen“ [Nr. 7]**

**Inhalt:** Schlechte Gliederung für Total-Anfänger (Tag 4: Reflection-Beispiel), Hauptaugenmerk liegt auf Applets (4).

**Layout:** Einfach langweilig (3).

**Änderungen:** Inhalt und Sprache, Übungen (4).

**V. Mukhi et al.: „Shlurrrpp . . . . . Java“ [Nr. 8]**

**Inhalt:** Keinerlei vernünftige Gliederung, viel zu oberflächlich, enthält wenig Informationen (5).

**Layout:** Überhaupt kein Layout . . . (4)

**Änderungen:** Inhalt und Sprache, Übungen (5).

**M. Pawlan: „Essentials of the Java Programming Language 1 + 2“ [Nr. 9]**

**Inhalt:** Zu wenig Hintergründe, manchmal etwas oberflächliche Informationen (2-).

**Layout:** Zu wenig Hervorhebungen, könnte ansprechender sein (2).

**Änderungen:** Zusammenfassungen, Übungen (3).

**D. Schmidt: „Programming Principles in Java“ [Nr. 10]**

**Inhalt:** Behandelt nur einige Themen, zu grobe Kapitel-Aufteilung (2).

**Layout:** Außer einigen Screenshots keinerlei Multimedia-Elemente, keine Links, überwiegend ASCII-Grafiken (4).

**Änderungen:** Mehr Multimedia-Elemente, besseres Layout, kleinere Kapitel (3).

**Abschließende Benotung**

Kriterium/Kurs	1	2	3	4	5	6	7	8	9	10
<b>Inhalt</b>	1	2	1+	3	3	2+	4	5	2-	2
<b>Layout</b>	1-	3	2	3	3	1	3	4	2	4
<b>Online-Kurs</b>	1	3	1-	4	3-	1-	4	5	3	3
<b>Gesamtnote</b>	1	3+	1-	3-	3	1-	4+	5+	2-	3

# Anhang B

## Dateien und Verzeichnisse

### B.1 Verzeichnisstruktur

Die Verzeichnisstruktur von iTeach kann bis zu einem gewissen Grad über eine zentrale Konfigurationsdatei beeinflusst werden; die folgende Aufstellung enthält darum die Namen der voreingestellten Verzeichnisse.

#### **iteach**

Enthält die beiden Haupt-Konfigurationsdateien: `iteach.xmap` enthält die Cocoon-Sitemap und `iteach.xconf` enthält ausführlich dokumentierte globale Konfigurationsparameter des Systems.

#### **iteach/docs**

Enthält u. a. die statischen XML-Dateien wie Online-Hilfe, Glossar, Login- und Stereotypen-Formular sowie die Datei `ontology.xml`, die die abstrakte Ebene des Domänenmodells beschreibt (s. Abschnitt B.2).

#### **iteach/docs/contents**

In diesem Verzeichnis finden sich für jedes Konzept eine XML-Datei `concept-id.xml`, die die Inhalte der jeweiligen Fragmente enthält (die Verknüpfung zu den Metadaten erfolgt über eine eindeutige Fragment-ID).

#### **iteach/docs/metadata**

Enthält für jedes Konzept eine XML-Datei `concept-id.xml`, die die Metadaten der jeweiligen Fragmente beschreibt (vgl. Abschnitt B.2).

#### **iteach/docs/tests**

Enthält die Dateien der Übungsaufgaben; in diesen XML-Dateien sind sowohl Inhalt als auch Verarbeitungsanweisungen enthalten.

#### **iteach/docs/xsp**

Dieses Verzeichnis enthält mehrere kleine XSP-Dateien, in erster Linie für Fehlermeldungen des Systems an den Benutzer (z. B. wenn ein Konzept oder Fragment mit unbekannter ID angefordert wurde).

**iteach/resources**

Die einzige Datei in diesem Verzeichnis ist `users.xml`, die eine Liste aller registrierter Benutzer (Login-Kennung und MD5-verschlüsseltes Passwort) enthält.

**iteach/resources/config**

Hier befinden sich weitere Konfigurationsdateien, u. a. für den `GlossaryTransformer` und für die Auswertung des Stereotypen-Formulars.

**iteach/resources/images**

Dieses Verzeichnis ist das Standardverzeichnis sämtlicher Grafiken. Buttons und Piktogramme finden sich hier, während Grafiken zum Inhalt im Unterverzeichnis `content` enthalten sind.

**iteach/resources/styles**

Hier liegen die CSS-Dateien mit verschiedenen Stil-Vorgaben; es existieren für den MS Internet Explorer und Netscape verschiedene Stylesheets - die Unterscheidung des Browsers geschieht über einen Cocoon Selector in der `iTeach-Sitemap`.

**iteach/stylesheets**

Enthält die beiden XSLT-Stylesheets `assembleHelpPage.xsl` und `assembleResultPage.xsl`, die für die Anzeige von Tests benötigt werden.

**iteach/stylesheets/html**

Enthält die für die Ausgabe in HTML erforderlichen Stylesheets.

**iteach/stylesheets/tests**

Enthält die für die verschiedenen Testtypen spezifischen Stylesheets.

## B.2 Formate der wichtigsten XML-Dateien

Exemplarisch soll hier die DTD von `ontology.xml` gezeigt werden:

```
<?xml version="1.0" encoding="UTF-8"?>
<!ELEMENT ontology (max-knowledge, node-set, linktype-set, edge-set)>
<!ATTLIST ontology
  name CDATA #REQUIRED>
<!ELEMENT max-knowledge (#PCDATA)>
<!ELEMENT node-set (node+)>
<!ELEMENT linktype-set (link-type+)>
<!ELEMENT edge-set (edge+)>

<!ELEMENT node EMPTY>
<!ATTLIST node
  id CDATA #REQUIRED
  name CDATA #REQUIRED
  start (false | true) #IMPLIED>
```

```
<!ELEMENT link-type EMPTY>
<!ATTLIST link-type
  name CDATA #REQUIRED
  required (false | true) #REQUIRED>

<!ELEMENT edge EMPTY>
<!ATTLIST edge
  from CDATA #REQUIRED
  to CDATA #REQUIRED
  name CDATA #REQUIRED
  priority CDATA #REQUIRED
  threshold CDATA #REQUIRED>
```

Die Metadaten sehen analog zum folgenden Datensatz zum Fragment „Hypertext“ des Konzepts „Hypermedia“ aus:

```
<fragment type="text" id="hm.ht">
  <title>Was ist Hypertext?</title>
  <description>Beschreibung des Begriffs Hypertext</description>
  <created>2001-07-25</created>
  <author>Axel Arne Guicking (mail@guicking.de)</author>
  <order-constraints>
    <constraint type="before" id="hm.mm"/>
  </order-constraints>
  <presentation-rules>
    <rule priority="2">
      <condition operator="or">
        <knowledge id="hm" comparator="greater">6</knowledge>
        <last-concept id="ah"/>
      </condition>
      <action>hide</action>
    </rule>
    <rule priority="1">
      <condition>
        <knowledge id="hm" comparator="greater-equal">2</knowledge>
      </condition>
      <action>shrink</action>
    </rule>
  </presentation-rules>
  <post-conditions>
    <concept main="true" id="hm" value="2"/>
  </post-conditions>
</fragment>
```

Für das Format der anderen Dateien sei auf diese selbst verwiesen.

# Anhang C

## Installation und Konfiguration

Die Installation von iTeach lässt sich prinzipiell in die folgenden Schritte zerlegen:

### 1. Installation eines Webservers

Als Webserver wurde während der Entwicklung Apache Tomcat 3.1 bzw. 3.2 eingesetzt. Auf der beiliegenden CD ist die Version 3.2.2 enthalten, es sind aber natürlich auch andere Webserver mit Servlet-Unterstützung (Version 2.2 oder höher) einsatzfähig.

Die Installation von Tomcat 3.2.2 besteht lediglich im Entpacken des ZIP-Archivs in ein beliebiges Verzeichnis (im folgenden `$tomcat-root$`). Über die Skripte `startup` und `shutdown` im Verzeichnis `$tomcat-root$/bin` kann der Server gestartet und beendet werden. Voraussetzung ist lediglich ein JRE (1.1 oder neuer) bzw. ein entsprechendes JDK.

### 2. Installation von Cocoon

Auch die Installation von Cocoon ist prinzipiell nicht weiter schwierig; die Vorgehensweise, insbesondere in Kombination mit Tomcat 3.2.2 ist in der Datei `install.html` im Cocoon-Paket beschrieben. Nach erfolgreicher Installation wird beim ersten Aufruf von Cocoon das WAR-Archiv `cocoon.war` in das Verzeichnis `$tomcat-root$/webapps/cocoon` entpackt.

### 3. Installation von iTeach

iTeach ist als „Teilpaket“ von Cocoon realisiert; dementsprechend müssen sich die `class`-Dateien im Classpath des Webservers und die restlichen Dateien für den „Web-Auftritt“ in einem Unterverzeichnis von `$tomcat-root$/webapps/cocoon` befinden. Dazu füge man die auf der CD befindliche Datei `iteach/lib/iteach-1.0.jar` dem Classpath hinzu (z. B. durch Kopieren in das Verzeichnis `$tomcat-root$/lib`) und entpacke die Datei `iteach/web/iteach-web-1.0.zip` in das Verzeichnis `$tomcat-root$/webapps/cocoon`, sodass ein Unterverzeichnis mit Namen `iteach` angelegt wird, das die Dateien des Archivs enthält.

#### 4. Anpassung von Cocoon und iTeach an die Systemgegebenheiten

Um iTeach zu konfigurieren, sind folgende Schritte erforderlich:

1. Hinzufügen der folgenden Sub-Sitemap-Definition in `$tomcat-root$/webapps/cocoon/cocoon.xmap` unter `map:pipelines`:

```
<map:pipeline>
  <map:match pattern="iteach/**">
    <map:mount uri-prefix="iteach"
      src="iteach/iteach.xmap"
      check-reload="yes"/>
  </map:match>
</map:pipeline>
```

2. Hinzufügen des folgenden Eintrags in `$tomcat-root$/webapps/cocoon/Web-inf/web.xml` (dient dem Laden des iTeach-StartupServlets bei Initialisierung von Cocoon):

```
<servlet>
  <servlet-name>iteach.startup</servlet-name>
  <servlet-class>de.iteach.StartupServlet</servlet-class>
  <load-on-startup>-2147483646</load-on-startup>
  <init-param>
    <param-name>config-file</param-name>
    <param-value>/iteach/iteach.xconf</param-value>
  </init-param>
</servlet>
```

Abschließend können bei Bedarf die Konfigurationsparameter und Einstellungen angepasst werden.

Die dieser Arbeit beigelegte CD-ROM enthält die folgenden Verzeichnisse:

##### **iteach**

Enthält alle das System iTeach betreffenden Dateien in den Unterverzeichnissen `lib`, `src` und `docs`, wie man es von anderen Java-Paket-Distributionen gewöhnt ist. Die schriftliche Ausarbeitung befindet sich zusammen mit der Quellcode-Dokumentation im Verzeichnis `docs`.

##### **packages**

Enthält die originalen Pakete (Tomcat 3.2.2, Cocoon 2.0b1 und Avalon 4.0b1). Avalon ist nur der Vollständigkeit halber enthalten, da es bei der Installation von Cocoon automatisch installiert wird.

# Literaturverzeichnis

- [1] BEAUMONT, I. H. User Modelling in the Interactive Anatomy Tutoring System ANATOM-TUTOR. *User Modeling and User-Adapted Interaction* 4, 1 (1994), S. 21–45. Nachdruck in [9], S. 91–115.
- [2] BEHME, H., AND MINTERT, S. *XML in der Praxis*. Addison-Wesley, München, 2000.
- [3] BLUMSTENGEL, A. *Entwicklung hypermedialer Lernsysteme*. Doktorarbeit, Universität Paderborn, 1998.  
→ <http://dsor.upb.de/de/forschung/publikationen/blumstengel-diss/>.
- [4] BOOCH, G., RUMBAUGH, J., AND JACOBSON, I. *Das UML-Benutzerhandbuch*. Addison-Wesley, Bonn, 1999.
- [5] BRUSILOVSKY, P. Methods and Techniques of Adaptive Hypermedia. *User Modeling and User-Adapted Interaction* 6, 2–3 (1996), S. 87–129. Nachdruck in [9], S. 1–43.  
→ <http://www.contrib.andrew.cmu.edu/~plb/UMUAI.ps>.
- [6] BRUSILOVSKY, P. Adaptive and Intelligent Technologies for Web-based Education. *Künstliche Intelligenz* 4 (1999), S. 19–25.  
→ <http://www2.sis.pitt.edu/~peterb/papers/KI-review.html>.
- [7] BRUSILOVSKY, P., AND EKLUND, J. A Study of User Model Based Link Annotation in Educational Hypermedia. *Journal of Universal Computer Science* 4, 4 (1998), S. 429–448.  
→ [http://www.jucs.org/jucs\\_4\\_4/a\\_study\\_of\\_user/paper.html](http://www.jucs.org/jucs_4_4/a_study_of_user/paper.html).
- [8] BRUSILOVSKY, P., EKLUND, J., AND SCHWARZ, E. Web-based education for all: A tool for developing adaptive courseware. *Computer Networks and ISDN Systems* 30, 1–7 (1998), S. 291–300.  
→ <http://www7.scu.edu.au/programme/fullpapers/1893/com1893.htm>.
- [9] BRUSILOVSKY, P., KOBSA, A., AND VASSILEVA, J., Hrsg. *Adaptive Hypertext and Hypermedia*. Kluwer Academic Publishers, Dordrecht, Niederlande, 1998.
- [10] BRUSILOVSKY, P., AND PESIN, L. Visual annotation of links in adaptive hypermedia. In *Proceedings of the CHI'95 Conference, Denver, CO (Mai 1995)*, S. 222–223.  
→ [http://www.acm.org/sigchi/chi95/Electronic/documnts/shortppr/plb\\_bdy.htm](http://www.acm.org/sigchi/chi95/Electronic/documnts/shortppr/plb_bdy.htm).

- [11] BRUSILOVSKY, P., SCHWARZ, E., AND WEBER, G. ELM-ART: An Intelligent Tutoring System on World Wide Web. In *Intelligent Tutoring Systems (Lecture Notes in Computer Science)*, C. Frasson, G. Gauthier, and A. Lesgold, Hrsg., vol. 1086. Springer-Verlag, Berlin, 1996, S. 261–269.  
→ <http://www.contrib.andrew.cmu.edu/~plb/ITS96.html>.
- [12] BUSH, V. As we may think. *The Atlantic Monthly* (Juli 1945).  
→ <http://ccat.sas.upenn.edu/jod/texts/vannevar.bush.html>.
- [13] CONLAN, O., AND WADE, V. Novel Components for supporting Adaptivity in Education Systems – Model-based Integration Approach. In *Proceedings of the ACM Multimedia Conference* (Los Angeles, CA, 2000).  
→ <http://www.acm.org/sigs/sigmm/MM2000/ep/conlan/index.html>.
- [14] DE BRA, P. Design Issues in Adaptive Web-Site Development. In *Proceedings of the Second Workshop on Adaptive Systems and User Modeling on the WWW* (Toronto and Banff, Canada, 1999), S. 29–39.  
→ <http://www.wis.win.tue.nl/asum99/debra/debra.html>.
- [15] DE BRA, P., AERTS, A., HOUBEN, G.-J., AND WU, H. Making General-Purpose Adaptive Hypermedia Work. In *Proceedings of the WebNet Conference* (2000), S. 117–123.  
→ <http://www.wis.win.tue.nl/~debra/webnet2000/debra.ps>.
- [16] DE BRA, P., AND CALVI, L. AHA! An open Adaptive Hypermedia Architecture. *The New Review of Hypermedia and Multimedia* 4 (1998), S. 115–139.  
→ <http://www.wis.win.tue.nl/~debra/review/paper.html>.
- [17] ECKEL, B. *Thinking in Java*, 2. Auflage. Prentice Hall, 2000.  
→ <http://www.mindview.net/Books/TIJ/>.
- [18] EKLUND, J., AND BRUSILOVSKY, P. The Value of Adaptivity in Hypermedia Learning Environments: A Short Review of Empirical Evidence. In *Proceedings of the Second Adaptive Hypertext and Hypermedia Workshop, AH'98* (Pittsburgh, PA, Juni 1998).  
→ <http://www.wis.win.tue.nl/ah98/Eklund.html>.
- [19] Microsoft Encarta Enzyklopädie, 1999.
- [20] FISCHER, G. Dynamische Präsentation für ein Java Tutorial. Diplomarbeit, Lehrstuhl für Informatik II, Universität Würzburg, 2001. In Arbeit.
- [21] FOWLER, M. *Refactoring. Wie Sie das Design vorhandener Software verbessern*. Addison-Wesley, München, 2000.
- [22] GAMMA, E., HELM, R., JOHNSON, R., AND VLISSIDES, J. *Entwurfsmuster. Elemente wiederverwendbarer objektorientierter Software*. Addison-Wesley, München, 1996.

- [23] GOOSSENS, M., MITTELBACH, F., AND SAMARIN, A. *Der L<sup>A</sup>T<sub>E</sub>X-Begleiter*. Addison-Wesley, Bonn, 1994.
- [24] GUPTA, P. L. Bist du drin? Lernen via Internet: So leicht geht das. *Audimax 02/03* (2001), S. 8–9.
- [25] HENZE, N. *Adaptive Hyperbooks: Adaptation for Project-Based Learning Resources*. Doktorarbeit, Universität Hannover, 2000.  
→ <http://www.kbs.uni-hannover.de/~henze/diss.pdf>.
- [26] HENZE, N., AND NEJDL, W. Adaptivity in the KBS Hyperbook System. In *Proceedings of the 2nd Workshop on Adaptive Systems and User Modeling on the WWW* (Toronto, Canada, Mai 1999).  
→ <http://www.kbs.uni-hannover.de/~henze/paperadaptivity/Henze.html>.
- [27] HOHL, H., BÖCKER, H.-D., AND GUNZENHÄUSER, R. Hypadapter: An Adaptive Hypertext System for Exploratory Learning and Programming. *User Modeling and User-Adapted Interaction* 6, 2–3 (1996), S. 131–156. Nachdruck in [9], S. 117–142.
- [28] JÖRNS, G. Das Bit bestimmt die Bildung. Artikel in Heise Telepolis vom 4. März 2001.  
→ <http://www.heise.de/tp/deutsch/inhalt/co/7053/1.html>.
- [29] KAPLAN, C., FENWICK, J., AND CHEN, J. Adaptive Hypertext Navigation Based On User Goals and Context. *User Modeling and User-Adapted Interaction* 3, 3 (1993), S. 193–220. Nachdruck in [9], S. 45–69.
- [30] KOBASA, A. User Modeling: Recent Work and Hazards. In *Adaptive User Interfaces: Principles and Practise*, M. Schneider-Hufschmidt, T. Kühme, and U. Malinowski, Hrsg. North Holland Elsevier, Amsterdam, Niederlande, 1993.  
→ <http://zeus.gmd.de/~kobsa/papers/1993-aur-kobsa.pdf>.
- [31] KOBASA, A., MÜLLER, D., AND NILL, A. KN-AHS: An Adaptive Hypertext Client of the User Modeling System BGP-MS. In *Proceedings of the Fourth International Conference on User Modeling* (Hyannis, MA, 1994), S. 99–105.  
→ <http://www.ics.uci.edu/~kobsa/papers/1994-UM94-kobsa.ps>.
- [32] KREUTZ, R. *Das Eden Hypertextsystem: Strukturierte und adaptive Lehrdokumente für das Internet*. Doktorarbeit, RWTH Aachen, 2000.  
→ [http://www.klinikum.rwth-aachen.de/WebPages/mib/cbt/papers/diss\\_rkreutz/diss\\_rkreutz.pdf](http://www.klinikum.rwth-aachen.de/WebPages/mib/cbt/papers/diss_rkreutz/diss_rkreutz.pdf).
- [33] KRÜGER, G. *Goto Java 2*, 2. Auflage. Addison Wesley, München, 2000.  
→ <http://www.javabuch.de/>.
- [34] MINNAMEIER, M. DaISy – Die Datenbank des InformationsSystems. Studienarbeit am Lehrstuhl für Informatik VI, Universität Würzburg, 2001.

- [35] MÜLLER, R., AND OTTMANN, T. The "Authoring on the Fly" System for Automated Recording and Replay of (Tele)presentations. *Sonderausgabe „Multimedia Authoring and Presentation Techniques“ of ACM/Springer Multimedia Systems Journal* 8, 3 (Mai 2000).  
→ <http://ad.informatik.uni-freiburg.de/mmgroup/aof/docs/publications/mueller0300.html.en>.
- [36] OTTMANN, T., AND WIDMAYER, P. *Algorithmen und Datenstrukturen*, 2. Auflage. Wissenschaftsverlag, Berlin, 1993.
- [37] PILAR DA SILVA, D., VAN DURM, R., DUVAL, E., AND OLIVIÉ, H. A Simple Model for Adaptive Courseware Navigation. In *WebNet'97 World Conference* (Toronto, Canada, November 1997).  
→ [http://www.wis.win.tue.nl/infwet97/proceedings/da\\_silva\\_2\\_full.html](http://www.wis.win.tue.nl/infwet97/proceedings/da_silva_2_full.html).
- [38] PILAR DA SILVA, D., VAN DURM, R., DUVAL, E., AND OLIVIÉ, H. Concepts and documents for adaptive educational hypermedia: a model and a prototype. In *Proceedings of the Second Workshop on Adaptive Hypertext and Hypermedia, HYPERTEXT'98* (Pittsburgh, USA, Juni 1998).  
→ <http://www.wis.win.tue.nl/ah98/Pilar/Pilar.html>.
- [39] RICH, E. User Modeling via Stereotypes. *Cognitive Science* 3 (1979), S. 329–354.
- [40] ROJAS, R., KNIPPING, L., RAFFEL, U., AND FRIEDLAND, G. Elektronische Kreide: Eine Java-Multimedia-Tafel für den Präsenz- und Fernunterricht. Tech. Rep. B-00-17, Institut für Informatik, FU Berlin, Oktober 2000.  
→ <http://kazan.inf.fu-berlin.de/echalk/docs/report001031.pdf>.
- [41] ROSS, J. L. On-Line But Off Course: A Wish List for Distance Educators. *International Electronic Journal For Leadership in Learning* 2, 3 (1998).  
→ [http://www.ucalgary.ca/~iejll/volume2/Ross2\\_3.html](http://www.ucalgary.ca/~iejll/volume2/Ross2_3.html).
- [42] RUSSELL, S., AND NORVIG, P. *Artificial Intelligence. A modern Approach*. Prentice Hall International, Inc., Englewood Cliffs, New Jersey, 1995.
- [43] SCHMUCKER, F. Navigation im Informationssystem. Studienarbeit am Lehrstuhl für Informatik VI, Universität Würzburg, 2001.
- [44] SCHULMEISTER, R. *Grundlagen hypermedialer Lehrsysteme*, 2. Auflage. R. Oldenbourg Verlag, München, Wien, 1997.
- [45] SCHULZ, S., ET AL. Qualitätskriterienkatalog für Elektronische Publikationen in der Medizin, 1999.  
→ <http://www.imbi.uni-freiburg.de/medinf/gmdsqc/d.htm>.

- [46] SEER, I. Thema: Elektronische Kreidetafel. Informatiker entwickeln elektronische Kreide: Multimedia-Tafel für den Präsenz- und Fernunterricht. TeachersNews.net Newsletter vom 31. Januar 2001.  
→ [http://www.teachersnews.net/newsletter/010105\\_29.htm](http://www.teachersnews.net/newsletter/010105_29.htm).
- [47] SPECHT, M. *Adaptive Methoden in computerbasierten Lehr/Lernsystemen*. Doktorarbeit, Universität Trier, 1998.  
→ <http://www.gmd.de/publications/research/1998/024/Text.pdf>.
- [48] STARKLOFF, P. WWWeiterbildung. *c't*, 4 (2001), S. 34.
- [49] VASSILEVA, J. A task-centered Approach for User Modeling in a Hypermedia Office Documentation System. *User Modeling and User-Adapted Interaction* 6, 2–3 (1996), S. 185–223. Nachdruck in [9], S. 209–247.
- [50] Die Virtuelle Hochschule Bayern.  
→ <http://www.vhb.org>.
- [51] WEBER, G., AND SPECHT, M. User modeling and Adaptive Navigation Support in WWW-based Tutoring Systems. In *Proceedings of the Sixth International Conference on User Modeling, UM97 (Sardinia, Italy)* (1997), S. 289–300.  
→ <http://www.cs.uni-sb.de/UM97/ps/WeberG.ps>.
- [52] WECKLEIN, D. Webbasierte Informationssysteme – Eine Oberfläche zur Suche. Studienarbeit am Lehrstuhl für Informatik VI, Universität Würzburg, 2001.

# Verzeichnis der Internetadressen

Die folgende Liste enthält die in Zusammenhang mit dieser Arbeit „verwendeten“ Adressen; die meisten von ihnen tauchen im Text in Form von Fußnoten auf. Es kann natürlich keine Garantie für die zukünftige Verfügbarkeit der Angaben übernommen werden, sie wurden jedoch alle auf ihre Korrektheit überprüft (Stand: 23. Juli 2001).

<http://ad.informatik.uni-freiburg.de/mmgroupp/aof/index.html.de>

„Authoring on the fly“-System zur Aufzeichnung von Vorlesungen o. ä.

<http://chortle.ccsu.ctstateu.edu/cs151/cs151java.html>

Introduction to Computer Science using Java von B. Kjell.

<http://d3web.informatik.uni-wuerzburg.de:8666/d3web/servlet/Index>

Eine der Infosystem-Homepages des Lehrstuhls für Informatik VI.

<http://edessa.topo.auth.gr/~thalis/java0.html>

Shlurrrpp . . . . . Java von V. Mukhi et al.

<http://jakarta.apache.org/avalon/index.html>

Das Avalon Framework der Apache Software Foundation.

<http://java.sun.com/developer/onlineTraining/Programming/index.html>

Essentials of the Java Programming Language 1 + 2 von Monica Pawlan.

<http://java.sun.com/docs/books/tutorial/index.html>

Das Java-Tutorial von Sun.

<http://java.sun.com/docs/white/langenv/index.html>

„The Java Language Environment – A White Paper“ von J. Gosling und H. McGilton.

<http://ltsc.ieee.org>

IEEE Learning Technology Standards Committee.

<http://math.hws.edu/javanotes/index.html>

„Introduction to Programming using Java“ von D. J. Eck.

<http://www.altavista.com>

WWW-Suchmaschine.

<http://www.apache.org>

Die Apache Software Foundation.

<http://www.aphorismen.de>

Sehr umfangreiche Sammlung von Aphorismen zu verschiedensten Themen. Aus dieser Sammlung stammen zum Großteil die Kapitel-einleitenden Zitate.

<http://www.ariadne-eu.org>

Ariadne-Projekt zur Standardisierung von Lehrressourcen im Internet.

<http://www.dublincore.org/documents/1999/07/02/dces/index.html>

Dublin Core Metadata Element Set, Version 1.1.

<http://www.mindview.net/Books/TIJ/index.html>

„Thinking in Java 2“ von Bruce Eckel.

<http://www.cis.ksu.edu/~schmidt/CIS200/index.html>

Programming Principles in Java von D. Schmidt.

<http://www.cyveillance.com/us/newsroom/pressr/000710.asp>

Informationen zum Wachstum des Internet.

<http://www.e-kreide.de>

System zur (Live-)Übertragung von Vorlesungen ins Internet.

<http://www.fernuni-hagen.de>

Fernuniversität Hagen.

<http://www.google.com>

WWW-Suchmaschine.

<http://www.imsproject.org>

IMS Global Learning Consortium.

<http://www.javabuch.de>

Guido Krügers „Goto Java 2“.

<http://www.junit.org>

Das JUnit Testing Framework.

<http://www.mut.de/media/buecher/Java2/index.htm>

„Java in 21 Tagen“ von Laura Lemay.

<http://www.searchenginewatch.com/reports/sizes.html>

Informationen zum Umfang von Suchmaschinen.

<http://www.vhb.org>

Virtuelle Hochschule Bayern.

<http://www.viror.de>

Virtuelle Hochschule Oberrhein.

<http://www.w3.org>

World Wide Web Consortium (W3C). Für die Entwicklung von Standards wie XML, HTML, XHTML u. v. a. m. verantwortlich.

<http://xml.apache.org/cocoon2/index.html>

Das Cocoon 2-Framework der Apache Software Foundation.

## Ein Wort des Dankes . . .

Diese Diplomarbeit wurde durch Prof. Dr. Frank Puppe ermöglicht, bei dem ich mich für die Bereitstellung von Ressourcen bedanken möchte. Ferner geht mein Dank an meinen Betreuer Dipl. Inform. Christian Betz, der mir bei Fragen und Problemen stets mit konkreten und spontanen Hilfestellungen zur Seite stand.

Bei Markus Minnameier bedanke ich mich für die geduldige Unterstützung bei der Integration von iTeach und dem Informationssystem. Bei Prof. Dr. Jürgen Wolff von Gutenberg und Gregor Fischer bedanke ich mich für die Möglichkeit, einen aktiven Beitrag zu einem Kurs der Virtuellen Hochschule Bayern leisten zu können.

Ferner möchte ich mich herzlich bei Thomas Renner für die vielen konstruktiven Gespräche, Anregungen und Tassen Kaffee auf seinem Balkon bedanken; er hat mir durch seine Gedanken zur Funktionalität eines adaptiven Lehrsystems sehr geholfen.

Last but not least gilt ein ganz besonderer Dank meinen Eltern, die mich während meines Studiums nicht nur finanziell unterstützt haben und dieser Arbeit durch die unzähligen Korrekturvorschläge und Anregungen viele sprachliche und semantische Ecken nahmen. Die Begeisterung meiner Mutter für ferne Inseln im Pazifik inspirierte mich zu dem Beispiel in Kapitel 3.3.3.

Hiermit versichere ich, die vorliegende Arbeit selbstständig und nur mit den angegebenen Hilfsmitteln erstellt und die Stellen, die anderen Werken dem Wortlaut oder dem Sinn nach entnommen sind, mit Quellenangaben kenntlich gemacht zu haben.

Würzburg, den 30. Juli 2001

Axel Arne Guicking