Kerres, Michael

Varianten computergestützten Instruktionsdesigns. Autorensysteme, Lehrprogrammgeneratoren, Ratgeber- und Konsultationssysteme

Unterrichtswissenschaft 24 (1996) 1, S. 68-92

Unterrichtswissenschaft

Zeitschrift für Lernforschung
24. Jahrgang / 1996 / Heft 1

Michael Kerres

Varianten computergestützten Instruktionsdesigns: Autorensysteme, Lehrprogrammgeneratoren, Ratgeber- und Konsultationssysteme.[1]

Variants of computer-based instructional design

Computer eignen sich nicht nur für die Durchführung, *sondern auch für* Planung *und* Entwicklung *von Unterricht. Die mangelnde Produktivität und didaktische Qualität bisheriger CBT-Produktionen motivierte Überlegungen zur „Automatisierung" von (Teilen) der didaktischen Planungs- und Entwicklungsaktivitäten.*
Eine weitgehende Automatisierung, insbesondere der Sequenzierung des Lehrangebotes, streben ID Expert sowie DISCourse an. Die Vorgehensweise der didaktischen Medienproduktion ändert sich bei diesen objektorientierten Entwicklungsumgebungen grundsätzlich, so daß Akzeptanzprobleme bei Instruktionsdesignern auftreten. Hinzu kommen eine Reihe grundsätzlicher technologischer wie didaktisch-konzeptueller Probleme, die zur Entwicklung von produktiven und analytischen Entwicklungswerkzeugen führte. Sie beschränken sich bewußt auf Teile des Entwicklungsprozesses, dienen jedoch zur Zeit eher der Ausbildung von Instruktionsdesignern.
Ratgebersysteme unterstützen die Planung und Entwicklung jeglicher Instruktion und lösen sich damit von computergestützten Lehr-Lernmedien. Elektronische Konferenz- und Konsultationsysteme schließlich fördern den persönlichen Austausch zwischen Experten über Distanzen.

Computers may be used for the delivery as well as the design and development of instruction. Current CBT-productions are being critizised for a lack of productivity as well as inferior didactic quality. This has motivated efforts to „automate" the design and development of instruction.
ID Expert as well as DISCourse aim at a „strong" automation, especially the sequencing of instruction. The object-oriented software engineering tools change the process of developing instructional media essentially which in turn might deteriorate acceptance by instructional designers. Technological as well as didactic problems have encouraged the development of tools with a relative limited scope. They support authors with special tasks in the design and development of courseware materials. Due to implementation problems they are restricted to the training of instructional designers.
Computer based coaches assist at designing any kind of instruction and, therefore, are not restricted to computer based instruction. Electronic conference and consultation systems finally foster the interpersonal communication between instructional design experts over distances.

[1] Eine Reihe von Diplomanden der Fachhochschule Furtwangen haben an den hier diskutierten Entwicklungen und Untersuchungen mitgewirkt. Ihr Beitrag ist im Text beschrieben. Wichtige Anregungen gaben darüber hinaus Jan Elen (Leuven, Belgien) und Begonia Gros (Barcelona, Spanien).

Mit dem Einsatz digitaler Rechner können Teilfunktionen von Lehr-Lern-Aktivitäten auf ein technisches System übertragen und damit *automatisiert* werden. Lange Zeit konzentrierte man sich auf die Frage, ob und wie Computer für die *Durchführung* von Unterricht nutzbar sind, sei es als computergestützter Unterricht, im Medienverbund oder zur Unterstützung personalen Unterrichtens. Zunehmendes Interesse findet jedoch die Diskussion des Computereinsatzes in *Unterrichtsplanung* und *-vorbereitung/-entwicklung,* dem computergestützten Instruktionsdesign (zur Diskussion unterschiedlicher Konzeptionen des Begriffs s. Lowyck & Elen, 1991; Schiffman, 1991).

Bei der Entwicklung computergestützter Lehr-Lernmedien (CBT) ist der Einsatz des Computers offensichtlich: „Autorensysteme" unterstützen die Entwicklung des Lehr-Lernmediums. Zusehends deutlicher werden die Probleme dieser Werkzeuge im Hinblick auf die *Produktivität* sowie die *didaktische Qualität* der so entwickelten CBT-Anwendungen. Dies führte zu Versuchen, den didaktischen Entscheidungsprozeß bei der CBT-Produktion in Software abzubilden, – in der Hoffnung, den didaktischen Entscheidungsprozeß selbst „automatisieren" zu können. Dies motivierte die Entwicklung von Werkzeugen, die sich nicht mehr auf die CBT-Entwicklung beschränken, sondern die Planung von Unterricht generell zu unterstützen versuchen.

Im folgenden werden diese verschiedene Varianten des computergestützten Instruktionsdesigns vorgestellt und Erfahrungen berichtet, die bei der Untersuchung vorliegender oder eigener Systeme gewonnen werden konnten.

1. Probleme konventioneller Autorensysteme

„Autorensysteme" unterstützen die Entwicklung computergestützter Lehr-Lernmedien, indem sie Routinen zur Verfügung stellen, die in CBT-Anwendungen typischerweise anzutreffen sind. Dabei stellt sich die Frage, ob bzw. wie deren Funktionalität sich an *didaktischen* Konzepten ausrichten sollte.

Als ein erstes Autorensystem mit größerer Verbreitung gilt PLATO (Programmed Logic for Automatic Teaching Operations), das seit 1960 an der University of Illinois entwickelt wurde (vgl. Fischer & Löthe, 1975): Der Autor definiert den Interaktionsraum des Lehr-Lernmediums, in dem Informationen präsentiert werden und zu Eingaben aufgefordert wird. In der Strukturierung des Kurses orientiert sich PLATO an den grundlegenden Operationen des *Computers,* der Eingabe, Verarbeitung und Ausgabe von Daten, nicht aber an didaktischen Konzepten des Lehrens und Lernens.

Einen anderen Ansatz verfolgte TICCIT (Time-shared computer controlled information television), das an der Brigham Young Universität in den 70er Jahren entwickelt wurde (Bunderson, 1974). Im Unterschied zu PLATO basierte es auf einem bestimmten Instruktionsmodell, nämlich Merrills „Component Display Theory (CDT)" (z.B. 1983). Das Autorensystem stellt eine aus dem Instruktionsmodell abgeleitete Funktionalität zur Verfügung, die wiederum auf einer speziell darauf abgestimmten Hardware basierte. Es

nimmt damit eine Vorstrukturierung der konzeptuellen Arbeit auf der Grundlage didaktischer Kategorien vor. Vom Autor wird erwartet, daß die grundlegenden Begriffe und Konzepte verstanden werden, um sie anwenden zu können.

Erfahrungen mit diesen Autorensystemen belegen, daß das didaktisch „nackte" Konzept von PLATO bei Autoren mehr Akzeptanz fand als das an ein didaktisches Modell gebundene TICCIT (s. Eyferth, 1974, Cleary, Mayes & Padham 1976, s.a. Issing, 1967; Pagliaro, 1983, Hawkridge, 1988). Hypothesen für den Erfolg der an PLATO orientierten Autorensysteme lauten:

- Das der Computerfunktionalität angelehnte Schema war im Denken der (bislang oft aus dem Computerbereich stammenden) Autoren bereits weitgehend verankert. Das Erlernen des Autorensystems fiel dieser Zielgruppe damit leichter.
- Autoren verfügen über mehr Freiraum in der Umsetzung eigener Ideen und müssen sich nicht an die Strukturen eines bestimmten didaktischen Konzepts halten.
- Möglicherweise hängen die Probleme von TICCIT auch mit dem zugrunde liegenden Instruktionsmodell zusammen. Lediglich die frühe (und mittlerweile weiterentwickelte) CDT (Merrill, 1987; Merrill, Lee & Jones, 1991) wurde seinerzeit einer konsequenten Implementation in einem Autorensystem unterzogen.

Heutige Autorensysteme[2] können ebenfalls der Tradition von PLATO zugeordnet werden. Verbesserte Funktionalität und gesteigerter Bedienungskomfort ermöglicht es CBT-Autoren, sich – ohne größere computertechnische Vorkenntnisse – ganz auf den didaktischen Gestaltungsprozeß zu konzentrieren. Heutige Autorensysteme verhalten sich damit scheinbar invariant gegenüber didaktischen Konzepten: Sie erlauben die Produktion „guter" wie „schlechter", theoretisch begründeter wie unbegründeter Lehr-Lernmedien gleichermaßen. Es ließe sich jedoch auch beklagen, daß keinerlei inhaltliche Unterstützung zur Entwicklung didaktisch wertvoller Materialien geboten wird.

Die in vielen Fällen bemängelte *didaktische Qualität* von CBT geht einher mit der weiterhin (zu) geringen *Produktivität* der didaktischen Medienentwicklung: Der Entwicklungsaufwand ist weiterhin so hoch, daß Qualitätskriterien oft zurückgestellt werden und die Entwicklung konzeptuell anspruchsvollerer Lehr-Lernmedien verhindert wird. Als derartige Mängel der Entwicklungswerkzeuge können etwa identifiziert werden (zur CBT-Produktion vgl. Ulloa, 1991; Boder & Gardiol Gutierrez, 1992; Kerres, 1993):

---

[2] Hypertext-Entwicklungssysteme werden im folgenden ebenfalls als – zunehmend konvergierende – Variante eines Autorensystems aufgefaßt, die sich für die Entwicklung von multimedialen Informationssystemen mit vernetzten Interaktionsräumen und damit weniger für stärker strukturierte Lehrprogramme eignen.

- Weitgehend lineare Anwendungen lassen sich unproblematisch entwikkeln, die komfortable Entwicklung komplexer Interaktionsräume mit vielen Verzweigungen wird jedoch kaum unterstützt z.B. durch angemessene Visualisierung „interaktiver Drehbücher". Dies führt zur Beibehaltung eher linearer Interaktionsräume. Die Orientierung an Instruktionsansätzen, wie sie konstruktivistische Lehr-Lerntheorien anbieten, wird kaum nahe gelegt oder gefördert (vgl. Ohlsson, 1993).
- "Fehler" bei interaktiven Drehbüchern (z.B. Sackgassen, unausgewogene Verzweigungsbäume/-ebenen, fehlende/übermäßige Rücksprünge, unangemessene Navigationsinstrumente etc.), sind – ab einer gewissen Komplexität – schwer identifizierbar und erhöhen den Test- und Wartungsaufwand.
- Ergebnisse vorgelagerter didaktischer Analysen können in der Entwicklungsphase nicht übernommen werden. Dies führt dazu, daß planerische und konzeptuelle Überlegungen möglicherweise vernachlässigt werden, nicht angemessen durchgeführt oder ungünstig umgesetzt werden. Die „Freiheit" der Autorensysteme legt nahe, daß die Arbeit des Autors mit der Eingabe von Daten beginnt und nicht mit didaktischen Erwägungen.
- Der Aufwand zur Anpassung der Anwendung an unterschiedliche Zielgruppen, Lehrgegenstände und -bedingungen ist unangemessen hoch: Die Autorin hat z.B. eine Anwendung entwickelt, die sich möglichst genau an bestimmten Gegebenheiten *einer* Zielgruppe orientiert, z.B. Anfänger mit hoher Lernmotivation. Die Anpassung an andere Zielgruppen erfordert i.a. eine komplette Reorganisation der Anwendung, deren Aufwand nicht selten einer Neuimplementation gleich kommt. Auch sachliche Änderungen oder Erweiterungen erfordern oftmals umfangreiche Modifikationen. Die Produktivität ist folglich durch mangelhafte Wartbarkeit einer vorhandenen Anwendung und schlechte Wiederverwertbarkeit in zukünftigen Anwendungen beeinträchtigt.
- Autorensysteme „behindern" das Management bereits mittelgroßer (z.B. verteilter) Anwendungen und vor allem die (übliche) gemeinsame Entwicklung von CBT-Anwendungen in einem Team (Dobson et al., 1993).

2. Anfänge der automatisierten Lehrprogrammerstellung

Ebenso wie die ersten Autorensysteme reichen auch Überlegungen zu deren Weiterentwicklung bis in die 60er Jahre zurück. *Bildungskybernetische Ansätze* versuchten die *Entwicklung* von Lehrprogrammen selbst zu systematisieren: ‚Algorithmisierung der Lehralgorithmen-Entwicklung" (Frank, 1966) beschreibt das seinerzeitige Anliegen.

Üblicherweise geschieht die Entwicklung eines Lehrprogramms weitgehend intuitiv. Wenn man diesen Vorgang systematisiert, in einen Algorithmus überführt und von einem Computer ausführen läßt, könnten Lehrprogramme automatisch generiert werden. Dies war die Idee einer „Formaldi-

daktik" von Helmar Frank. Er unterscheidet zwischen voll objektivierten und halbalgorithmischen Didaktiken, bei denen das Lehrprogramm nur teilweise vom Rechner generiert wird. Ludwig (1965) und Zifreund (1965) stellten systematische Verfahren zur Entwicklung von Lehrprogrammen vor.

Als der erste automatisierte Programmgenerator kann ALZUDI (algorithmischen Zuordnungdidaktik) gelten, der zwischen 1966 und 1969 an der PH Berlin entwickelt wurde (Frank & Graf, 1967): Ziel war die „Zuordnung" von deutschen Vokabeln zu fremdsprachigen Ausdrücken, wobei das Programm für beliebige Vokabeln einen Lochstreifen für ein Wiedergabegerät und damit ein Lehrprogramm generierte.

Mit dem späteren Programmgenerator ALSKINDI (Algorithmische Skinnerstil-Didaktik) werden Lehrprogramme auf der Grundlage von „Basaltextsätzen" mit „Basalwörtern" erzeugt (Frank, 1969). „Basalwörter" sind die grundlegenden, redundanzfreien Informationseinheiten eines Textes, die die zu vermittelnden Konzepte eines Lehrprogramms anzeigen. Sie sind in knapper Form in „Basaltextsätze" einzukleiden. Zu den „Basalwörtern" anzugeben sind jeweils Vorkenntnis- und Sollwertzahlen (Kenntnisswahrscheinlichkeit vor bzw. nach Bearbeitung des Lehrprogramms). Das durch ALSKINDI erzeugte Programm präsentiert zunächst einen „Basaltextsatz". Anschließend werden einzelne „Basalwörter" dieses Satzes ausgeblendet und abgefragt bis ein bestimmtes Kenntnisniveau erreicht ist. Es folgt die Präsentation des nächsten Basaltextsatzes (vgl. Frank, 1969).

Mit ALSKINDI war das Konzept einer Trennung von Lehrinhalts- und Lehrstrategie-Komponente erstmalig formuliert und realisiert: Ein Modul beinhaltet das zu vermittelnde Wissen, ein anderes Modul die Lehrstrategie. So würde etwa ein Sachexperte lediglich – in bestimmter Weise strukturiertes – Wissen in eine Lehrinhalts-Komponente eingeben müssen; das Lehrprogramm wird dann automatisch erzeugt.

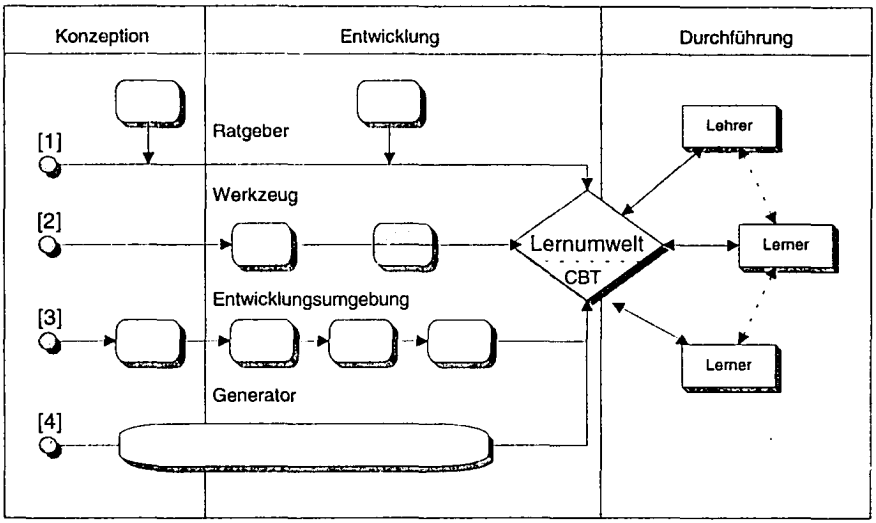
Frank (1969) sah bereits das grundlegende Problem der Generierung von instruktional verwendbaren Varianten auf der Grundlage von „Basalwörtern" und „Basaltextsätzen": "Alle Formaldidaktiken stehen vor der Schwierigkeit, daß man noch keinen Algorithmus kennt, denselben (z.B. im Basaltext formulierten) Sachverhalt mit anderen Worten, insbesondere mit größerer Redundanz, nochmals auszudrücken." Unabhängig von der gewählten Art der Wissensrepräsentation (sei es als Liste von Basalwörter oder als semantisches Netz, Objekthierarchie, Regel- oder Fallbasis etc., vgl. Wenger, 1987; Mandl, Hron & Tergan, 1990, zu neueren Ansätzen) muß ein abstrahiertes Wissen für ein Lehrprogramm in besonderer Weise aufbereitet, didaktisch transformiert, werden. Diese Hürde ist bis heute nicht überwunden.

3. Aktuelle Ansätze des computergestützten Instruktionsdesigns

Ziel des Instruktionsdesigns sei die Planung und Entwicklung einer Lernumgebung (mit Bestimmung von Lehrinhalten, Lehrzielen, Lernaufgaben und

-materialien etc., vgl. z.B. Seel, 1981, 1991). Computergestützte Lehr-Lernsysteme sind *ein möglicher Teil* dieser Lernumgebung: Lerner interagieren u.a. mit Lehrern, anderen Lernern in bzw. mit einer didaktisch strukturierten Lernumgebung (vgl. Tennyson, Spector & Muraida, 1994; Wasson, 1994).

Abbildung 1: Ansätze des computergestützten Instruktionsdesigns



Alle Bemühungen eines computergestützten Instruktionsdesigns verfolgen vorrangig die Ziele: Steigerung der *Produktivität* durch Reduktion des Entwicklungsaufwands und/oder Steigerung der *didaktischen Qualität* der resultierenden Lernumgebungen. Dabei können folgende Ansätze unterschieden werden:

- die weitgehende Automatisierung der CBT-Entwicklung auf der Grundlage allgemeiner Spezifikationen im Sinne generierender Systeme (s. [4] in Abb. 1): Ziel sind komplette, leicht adaptierbare CBT-Anwendungen,
- die Automatisierung bestimmter Teile der CBT-Entwicklung im Sinne eines *Computer Aided Software Engineering* (s. [3] in Abb. 1) durch Entwicklungsumgebungen, die (möglichst) den gesamten Entwicklungsprozeß abbilden, dem Autor aber gestaltende Eingriffe ermöglichen,
- die Unterstützung der Instruktionsentwicklung durch analytische und produktive Werkzeuge (sowohl für CBT als auch für andere Lehr-Lernmedien bzw. Bestandteile der Lernumgebung) (s. [2] in Abb. 1), die sich auf *Teile* des Entwicklungsprozeß beschränken,
- die Unterstützung des Instruktionsdesigns durch didaktische Ratgebersysteme unabhängig von den Merkmalen der zu konkretisierenden Lernumgebung (s. [1] in Abb. 1): Ergebnis sind textliche Hinweise für Instruk-

tionsdesign und -entwicklung. Die Umsetzung in konkrete Medien bzw. Instruktion wird nicht automatisiert.

- die Unterstützung des Instruktionsdesigns durch elektronische Konferenz- und Konsultationssysteme. Der Computereinsatz dient hierbei vorrangig der Unterstützung *interpersoneller* Kommunikation.

Goodyear (1994) unterscheidet in diesem Zusammenhang zwischen *weak* und *strong automation,* je nach Grad und Intensität der angestrebten bzw. erzielten Automatisierung.

Weniger reflektiert wurde bislang der Computereinsatz bei *Bildungsplanung und -mangement* als möglicherweise produktivitätssteigernde Maßnahme (wie z.B. computergestützte Bedarfsermittlung, Kursanmeldung und -zuweisung, Evaluation und Dozentenrückmeldung, Kurs-Distribution sowie Prüfungsdurchführung, -auswertung und -rückmeldung vgl. dazu etwa Wulfeck et al., 1993).

4. Lehrprogrammgeneratoren und Entwicklungsumgebungen

4.1 Funktionsprinzip

Lehrprogrammgeneratoren versuchen die CBT-Entwicklung weitgehend zu automatisieren; Autoren soll insbesondere die Entscheidung über die Unterrichtssequenzierung abgenommen werden. Damit ähnelt der Anspruch derartiger Generatoren früheren Ansätzen „intelligenter tutorieller Systeme" (ITS) (vgl. z.B. Wenger, 1987). Während man bei ITS eine *mikro-adaptive* Sequenzierung (d.h. Lernregulation) anstrebte, also eine Anpassung auf Grundlage des laufenden Dialogs zwischen Lerner und System, handelt es sich hier um eine *makro-adaptive* Sequenzierung (d.h. Lernsteuerung) auf Grundlage zu Beginn festgelegter Merkmale des Lehr-Lernprozesses.

Dabei ist eine *grundlegende* technische Bedingung zu erfüllen: Eine Kurs darf nicht „explizit" formuliert werden (wie bei Autorensystemen), sondern ist in kleine Einheiten zu zerlegen und mit Deskriptoren zu versehen, auf die der Sequenzierungsalgorithmus beim Zusammensetzen des Kurses zugreift. Das System wird in Komponenten aufgeteilt: es existiert ein Mechanismus, der die Sequenzierung des Lehrangebotes übernimmt, und eine Datenbank, die die – auf bestimmte Weise strukturierten – Lehr-Lerneinheiten speichert.

Es werden dazu üblicherweise „kleinste instruktionale Einheiten" gebildet (z.B. die „Definition eines Mittelwertes als textliche Beschreibung"; ein „Beispiel für die Berechnung des geometrischen Mittelwertes"; eine weiteres „ausführlicheres Beispiel" hierfür; eine „Graphik, die dies erläutert"; ein „Tondokument, das die Definition vorliest" usw.). Zusätzlich werden Informationen („didaktische Deskriptoren") zu diesen „Einheiten" abgespeichert, die der generierende Algorithmus für die Sequenzierungsentscheidung benötigt, z.B. das Thema, das die Einheit behandelt, der gewählte

Schwierigkeitsgrad, die Art der Repräsentation, das vorausgesetzte Vorwissen.

Denn aus einer Menge von „Seiten", z.B. aus einem Lehrbuch (mit erläuternden Texten, Grafiken, Fragen etc.), kann ein Generator keine didaktisch begründete Sequenzierung vornehmen, da das Programm nicht „weiß", was auf der Seite steht. Er benötigt zusätzlich abstrakt gespeicherte Informationen zu der einzelnen Seite, z.B. behandeltes Thema, Art der Darstellung, Schwierigkeitsgrad der Darstellung etc. Das didaktisch-konzeptuelle ebenso wie technisch zu bewältigende Problem besteht in der „Zusammensetzung" eines Lehrangebotes aus den Informationen der unterschiedlichen Komponenten, der Sequenzierung des Lehrangebotes.

Der Entscheidungsalgorithmus beschreibt abstrakt, was „optimale" Kurssequenzen sind und wie Kurse grundsätzlich „zusammenzusetzen" sind. Aus unterrichtswissenschaftlicher Sicht stellt sich das Problem, welche Informationen benötigt werden, um „kleinste instruktionale Einheiten" zu einer Unterrichtseinheit zusammenzusetzen, und: was sind „optimale" Sequenzen solcher „instruktionaler Einheiten"?

Der Unterschied zu „intelligenten tutoriellen Systemen", die auf einer „Wissensbasis" operieren, wird deutlich. Eine „Wissensbasis" beinhaltet eine *abstrahierte* Repräsentation eines Sachgebiets (z.B. in Form von semantischen Netzen oder Wenn-Dann-Regeln), auf die eine tutorielle Komponente zugreift und instruktionelle Elemente zu generieren versucht (also z.B. textuelle Erläuterungen, Fragen, Rückmeldungen). Bei den hier diskutierten Systemen liegen die eigentlichen instruktionalen Einheiten jedoch bereits vor, generiert wird (nur) *die Sequenz* dieser Einheiten für z.B. verschiedene Zielgruppen.

4.2 Vorgehensweise

Unter dem Schlagwort „Automatisierung des Instruktionsdesigns" sind in den letzten Jahren eine Vielzahl von Prototypen entwickelt worden, die Teilaspekte des dargestellten Ansatzes untersuchten und auf ihre Brauchbarkeit prüften. Zu nennen wären etwa Xerox IDE, Russel, Morgan & Jordan, 1988; ECAL, Elsom-Cook & O'Mallay, 1989; SHIVA, Baker, 1994; Expert CML, Jones & Wipond, 1991; IDiOM, Gustafson & Reeves, 1990; SOCRATES, Ranker & Doucet, 1990; PEPE, Wasson, 1992 (s.a, 1994), (s.a. Perez & Seidel, 1990). Die folgende Darstellung beruht auf Untersuchungen von (Teilen von) ID Expert und DISCourse, die zur Zeit als am weitesten fortgeschrittenen Ansätze gelten können. ID Expert ist seit 1987 in der Entwicklung und basiert auf den Arbeiten von David Merrill. Es kommt am ehesten dem Konzept eines Programmgenerators nahe. Die Entwicklungsumgebung des Projektes „DISCourse" wurde im Rahmen des EU-Programms DELTA (Development of European Learning through Technological Advance), Vorläuferprojekte beginnend ab 1988, entwickelt. Bei beiden Ansätzen ist der Lehrgegenstand zunächst in eine abstrakte Struktur zu transformieren, d.h. es wer-

den nicht „Seiten" oder konkrete Bestandteile des Lehr-Lernmediums formuliert, sondern der Lehrgegenstand ist in Form einer Hierarchie zu bringen, in der Begriffe und deren Unter- zw. Überordnung dargestellt sind. Es wird damit gefordert, den Lehrgegenstand zunächst in abstrakter Form zu durchdringen. Die zu entwickelnde Hierarchie bezieht sich auf die *sach-logische Struktur,* nicht auf eine bereits didaktisch aufbereitete Struktur etwa im Sinne von Gagnés Lernhierarchien.

Bei ID Expert wird zusätzlich eine Sammlung konkreten Lehrmaterials in einer „Resourcenbank" angelegt. Bei DISCourse wird zunächst die „Domäne" modelliert, die die thematische Struktur des Lehrgebietes darstellt. Konkrete instruktionale Einheiten werden als „media units" den Themen zugeordnet und mit beschreibenden Elementen zu deren didaktischen Funktion verknüpft (Anekdote, Gegenbeispiel, Konzeptdefinition, Fallbeispiel, Illustration, Gedächtnishilfe, Einführung, Zusammenfassung usw., s. Chiocariello, Innocenti, Persico, Sarti, Viarengo, & van Marcke, 1993, die sich insbesondere auf Shuell, 1988, berufen). In einem weiteren Modul werden Lehrziele spezifiziert. Dabei fordert ID Expert die Einhaltung des Rasters von Merrills „Component Design Theory" (1987). Bei DISCourse werden Lehrziele im Klartext eingegeben und beginnend von übergeordneten Zielen zunehmend verfeinert (nach Posner & Rudnitsky, 1986, vgl. Baker, 1994). Als nächster Schritt werden Objekte der Domäne mit Lehrzielen verknüpft. Es werden nur die Lehrinhalte in einen Kurs übernommen, zu denen ein Lehrziel zugewiesen wurde. Dieser Vorgang der Zuordnung von Inhalten zu Lehrzielen definiert den eigentlichen Lehrstoff *einer* spezifischen CBT-Anwendung in dem „Content-Modul". Aus einer Sachgebiets-Domäne können so mehrere unterschiedliche CBT-Anwendungen entstehen.

Technisch weniger anspruchsvoll ist das Modul, das der Bestimmung von Merkmalen der Lehr-Lernsituation dient. Beide Ansätze beschränken sich hierbei auf wenige Attribute des Lerners (üblicherweise in binärer Ausprägung): Motivation, Vorwissen, Lernstil und Lernerkontrolle. (In unterschiedlichen Versionen von ID Expert und DISCourse sind auch andere Attribute zu finden.) Es ist möglich „Lernertypen" abzuspeichern, die durch eine bestimmte Konfiguration dieser Attribute charakterisiert werden.

Aus diesen instruktionalen Einheiten wird nun eine Instruktionssequenz zusammengesetzt. Besonderer Aufmerksamkeit gilt damit dem Mechanismus, der diese Sequenzierung vornimmt. Hierbei gehen ID Expert und DISCourse unterschiedliche Wege. Bei ID Expert wählt der Autor eine Transaktions-Shell, die die grundlegende Instruktionsstrategie festlegt. Verfügbar ist die erste, eher einfache Transaktions-Shell, die das „Identifizieren" und „Zuordnen" von Objekten zu anderen Objekten, z.B. die Positionierung von Städtenamen auf einer Landkarte, erlaubt. Zusätzlich sind neben dem Lehrgegenstand lediglich die gewünschte Zielgruppe auszuwählen. Das Programm generiert den Kurs.

Die Sequenzierungsentscheidungen des Systems bleiben „verborgen". Es kann nicht abfragt werden, welche Konsequenz bestimmte Einstellungen,

etwa „Lernmotivation: niedrig", auf den generierten Kursverlauf haben, auch eine Beeinflussung oder gar Veränderung ist ausgeschlossen. Dies entspricht dem Anspruch eines stark automatisierenden CBT-Generators.

Als Sequenzierungsstrategie wird im Rahmen von DISCourse GTE, das *generic tutoring environment,* von van Marke (1992) eingesetzt, das „generische", also von konkreten Lehrinhalten unabhängige Instruktionsstrategien beinhaltet. Es greift dabei auf die anderen Module zu und erzeugt – in Abhängigkeit von dem gewählten Lernerprofil – verschiedene Sequenzen instruktionaler Einheiten (s. Wasson, 1992, McIntyre, 1993). Aufgrund technischer Probleme[3] ist gerade dieser Schritt bis heute nur in Ansätzen erkennbar. Die Benutzeroberfläche behindert die genauere Analyse oder gar Modifikation und Erweiterung des Mechanismus. Gleichwohl muß als entscheidende Leistung gesehen werden, daß aufgezeigt wurde, wie (und daß) ein solcher *transparenter Mechanismus* implementiert werden kann, bei dem die Lehrstrategie sichtbar und modifizierbar ist.

4.3 Diskussion

DISCourse und ID Expert verfolgen ähnlich anspruchsvolle Ziele einer „starken" Automatisierung von Instruktionsdesign und -entwicklung für CBT, in der Implementation bestehen jedoch grundsätzliche Unterschiede. ID Expert basiert stärker auf einem ausformulierten theoretischen Raster und fordert deswegen Spezifikationen, die an diesen Konzepten auszurichten sind. Bei DISCourse macht sich – als europäischem Entwicklungsprojekt – die Aufteilung in verschiedene Teilprojekte bemerkbar, die eine Festlegung auf einen bestimmten Ansatz verhinderte (mit entsprechenden Vor- und Nachteilen).

Die didaktische Qualität der generierten Kurse muß zum jetzigen Zeitpunkt als deutlich eingeschränkt beurteilt werden. Die Kurse zeichnen sich durch ein hohes Maß an Standardisierung aus und bleiben einfachen expositorischen Instruktionsansätzen verhaftet. Die Qualität der Anwendung bleibt bisher (zwar zuverlässig, aber) immer auf einem relativ niedrigen Niveau, bei konventionellen Autorensystemen besteht zumindest die Möglichkeit zu hochwertigeren Konzeptionen.

Doch ist diese Frage immer im Verhältnis zu dem erreichbaren Produktivitätsgewinn zu diskutieren. Wittmann (1995) erstellte in ihrer Diplomarbeit eine der ersten Applikation mit der Entwicklungsumgebung von DISCourse, um die Auswirkungen auf die Vorgehensweise bei der CBT-Produktion zu untersuchen. Es zeigt sich, daß die objektorientierte Entwicklungsumge-

---

[3] GTE ist auf einer anderen Rechnerplattform implementiert als die anderen Teile von DISCourse und liegt bis dato lediglich als LISP-Codierung vor, die den Eingriff des Autors praktisch unmöglich macht.

bung von DISCourse die Vorgehensweise der CBT-Entwicklung entscheidend verändert:

- Die Eingaben sind in verschiedenen Modulen vorzunehmen: Benötigt wird ein tieferes Verständnis über das Zusammenwirken dieser Komponenten.
- Für die Arbeit mit dem Domänen- und Content-Modul werden Kenntnisse der Konzepte objektorientierter Softwareentwicklung benötigt.
- Das System fordert eine intensive Auseinandersetzung mit dem Lehrgegenstand, insbesondere dessen systematische Strukturierung, als Voraussetzung für die Eingabe in der objektorientierten Datenbank.
- Der Aufwand zur Entwicklung einer *einzelnen* Kurseinheit CBT ist höher als bei konventionellen Autorensystemen. Die Vorteile des Vorgehens können erst bei umfangreichen CBT-Produktionen (bei Modifikationen, Erweiterungen, Adaptionen oder Portierungen großer Anwendungen) sichtbar werden.

Die Auswirkungen dieser Vorgehensweise sind für unterschiedliche Benutzergruppen getrennt zu diskutieren:

(1) „Anfänger" werden eine weitgehende Automatisierung zunächst schätzen, da sie sich eine Reduktion des Arbeitsaufwandes erhoffen. Gleichwohl erhöht sich der Aufwand zum Verstehen des Arbeitsprinzips der Entwicklungsumgebung. Die eigene Lernmöglichkeit (im Sinne des Aufbaus didaktischer Expertise) durch das praktische Arbeiten an konkreten CBT-Projekten wird eingeschränkt, da die Kurssequenz automatisch generiert wird.

Ein Produktivitätsgewinn ist auch bei Anfängern nur zu erreichen, wenn die Komplexität der Entwicklungsumgebung beherrscht wird. Dies erfordert mehr Zeit als in anderen Systemen. Insofern ist nicht auszuschließen, daß bei gleicher Einarbeitungszeit in anderen Systemen (konventionellen Autorensystemen) qualitativ bessere Resultate erzielt werden können.

(2) Professionelle Instruktionsdesigner in CBT-Produktionsbetrieben müssen wesentlich umdenken und bemängeln die Einschränkung ihres Gestaltungsspielraums. Die Vorteile des alternativen Vorgehens werden nicht unmittelbar einsichtig. Erste Untersuchungen zeigen, daß gerade in dieser Gruppe mangelnde Akzeptanz vorherrscht (vgl. Baker, 1994, S. 219, Tait, 1992, S. 140). So erscheint eine stärkere Orientierung an der Vorgehensweise dieser Gruppe zwingend notwendig (vgl. Reimann, 1993, S. 265).

CBT-Produktionen werden bislang üblicherweise als *einzelne Projekte* betrachtet, die es möglichst effizient abzuwickeln gilt. Bei dieser Sichtweise bieten die neuen Werkzeuge keinen Vorteil. Betrachtet man dagegen CBT organisationsweit als Teil von Maßnahmen zur Einrichtung von „Lernumwelten" (vgl. Kerres, 1995), ergibt sich möglicherweise eine andere Einschätzung. Größere Aufgeschlossenheit ist folglich bei Instruktionsdesignern zu erwarten, die große CBT-Kurs"systeme" zu managen haben, und derartige,

z.B. (über Deutschland oder Europa verteilte) Anwendungen planen sowie praktische Erfahrungen mit der Schwierigkeit gemacht haben, dies mit konventionellen Methoden zu realisieren.

(3) Vor allem ID Expert setzt in den USA stark auf die Zielgruppe der „Sachexperten", die in Organisationen arbeiten und CBT nebenher entwickeln, ohne eigentliches Interesse an Fragen des Instruktionsdesigns. Unsicher erscheint uns das Potential dieser Zielgruppe in Europa. Vermutlich müssen hier kulturelle Unterschiede berücksichtigt werden (vgl. Kerres, 1991). Eine Steigerung der Produktivität entsteht nur, wenn (abgesehen von der Einarbeitungszeit) die Entwicklungsdauer eines Kurses bei vergleichbarer didaktischer Qualität geringer oder die Qualität der Anwendung bei einer gegebenen Entwicklungsdauer deutlich höher wäre als bei anderen Systemen. Beides ist mit den vorliegenden Systemen kaum zu erzielen.

Die Kosten für die Sachexpertin sind darüber hinaus sicherlich nicht niedriger als für die Instruktionsdesignerin, auch die Verfügbarkeit didaktischer Expertise ist nicht das Problem. Insofern erscheint die Zielgruppe der Sachexperten zumindest in Deutschland weniger relevant zu sein.

Betrachten wir als nächstes einige konzeptuelle Fragen, die den Aufbau der Systeme betreffen. So stellt sich (wie bei „intelligenten tutoriellen Systemen") die Frage der Trennbarkeit von *Lehrinhalt* und generischen *Lehrstrategien.* "Unterrichtsmethodik" kann zwar durchweg unabhängig von konkreten Lehrinhalten konzeptualisiert werden, zu diskutieren ist jedoch, ob konkreter Unterricht tatsächlich aus der bloßen *Verknüpfung* von „Inhalten" und „generischen Lehrstrategien" entsteht: Unterricht = Lehrinhalt + Lehrstrategie.

Aus didaktischer Sicht entsteht Unterricht jedoch erst durch Anwendung unterrichtsmethodischer Überlegungen, um Lehrinhalte in unterrichtliche Aktivitäten zu *transformieren* (vgl. die Kritik von Goodyear, 1994). Vernachlässigt wird also die Unterscheidung und Beziehung von sachlogisch strukturiertem Lehrgegenstand und – im Hinblick auf bestimmte Methoden entwickelte – instruktionale Elemente (vgl. den Begriff „Operationsobjekt" bei König & Riedel, 1979, s.a. Seel, 1981).

Zu problematisieren ist ebenso die Zuordnung von Lehrzielen zu „kleinsten instruktionalen Einheiten", der in beiden Ansätzen eine zentrale Bedeutung zukommt. Denn durch die Auflösung sinnhafter Einheiten, also etwa einer Unterrichtseinheit in „kleinste instruktionale Einheiten" verliert die Zuordnung von Lehrzielen ihre Aussagekraft. Es bleibt fraglich, ob sich einer Grafik, einem Textabschnitt oder einem Videoclip als solches ein Lehrziel oder eine Lernfunktion zuordnen läßt. Beide Systeme basieren jedoch wesentlich auf derartigen Zuordnungen.

Ein großer Teil der Komplexität der Entwicklungsumgebung resultiert aus dem Anspruch, das Lehrangebot „automatisch" sequenzieren zu können. Es stellt sich die Frage, ob der zusätzliche Aufwand, der dadurch bei der Entwicklung notwendig wird, angemessen ist. Zwei Aspekte wären zu beden-

ken: welche Bedeutung hat die Sequenzierung von Lehrangeboten für Autoren einerseits und für Lerner andererseits?

Auch wenn gefragt werden kann, ob die gewählten Sequenzierungen immer angemessen sind, gibt es keine Hinweise darauf, daß Autoren besondere Probleme mit der Sequenzierung von Lehrangeboten hätten und entsprechenden Unterstützungsbedarf äußern würden.

Die Bedeutung eines sequenzierten Lehrangebotes *für Lerner* verweist wiederum auf die aktuelle Konstruktivismus-Debatte (vgl. Jonassen, 1991): Der Lerner sollte danach unterstützt werden, einen angemessenen Lernweg möglichst eigenständig zu finden statt auf eine externe Steuerung bzw. einen externen Regler zu vertrauen.

Die Diskussion über Lerner- vs. Programmkontrolle in interaktiven Lehr-Lernmedien zeigt, daß eine Reihe von Konstellationen existieren, die durchaus für Programmkontrolle sprechen (vgl. Steinberg, 1989). Dennoch wäre anzustreben, daß die Systeme die Programmkontrolle zugunsten der Lernerkontrolle *variabel* gestalten können. Bei Lernerkontrolle wird die Sequenzierung des Lehrangebotes dann zunehmend dem Lerner überlassen. Insofern erschwert die Fixierung auf das Problem der automatischen (= programmkontrollierten) Sequenzierung den Übergang zu konstruktivistischen Instruktionsansätzen.

4.4 Fazit

Die bisherigen Erfahrungen zeigen, daß die Systeme keine Alternative für die Standard-CBT-Produktion darstellen. Es wäre falsch, aus dieser Kritik auf die Unbrauchbarkeit entsprechender Ansätze zu schließen, da bisherige Untersuchungen im Prinzip auf solche Beispielanwendungen und -kontexte beschränkt bleiben, für die sich diese Systeme möglicherweise eben gerade nicht eignen. Insofern wurden die Systeme bislang vermutlich unter Bedingungen untersucht, die deren Nutzen „unsichtbar" machen. Eine Untersuchung unter realen Bedingungen bei Entwicklung und Management umfangreicher CBT-Pools ist jedoch zum jetzigen Zeitpunkt nicht möglich (zur Problematik der Evaluation automatisierten Instruktionsdesigns vgl. Gros, 1994).

Es bleibt die Kritik, daß bislang nicht genügend reflektiert worden ist, für wen und was sich diese Systeme denn genau eignen (werden). Kritisch erscheint damit eher, daß an den Instruktionsdesignern „vorbei" entwickelt wurde, d.h. die Berücksichtigung deren Vorgehen fand nicht statt. Dies erfordert die Integration neuer Werkzeuge in die übliche Produktionsumgebung von Autoren und damit die Entwicklung von Schnittstellen zu Werkzeugen wie Toolbook oder Authorware (vgl. Ulloa, 1994). Statt einer Alternative zu Autorensystemen wäre der schrittweise Übergang zu einer übergeordneten Entwicklungsumgebung anzubieten.

5. Unterstützungs-Werkeuge für Autoren

Aus technischer Sicht weniger anspruchsvoll ist die Konstruktion von *Unterstützungs-Werkzeugen* für Instruktionsdesigner. Zu unterscheiden sind *produktive* Werkzeuge, die bestimmte (eingeschränkte) Teile der Medienentwicklung übernehmen sowie *analytische* Werkzeuge, die der Prüfung der Entwicklungsarbeit dienen.

Müller & Schweitzer (1994) entwickelten in ihrer Diplomarbeit ein produktives Werkzeug für die Entwicklung von Kursen mit Toolbook. Es wurde nicht angestrebt, Kurse zu „generieren", sondern eine Grundstruktur für den Interaktionsraum eines computergestützten Lehr-Lernmediums „vorzufabrizieren" (engl. „half-fabricates") – in Abhängigkeit von Merkmalen des Lehr-Lernprozeß.

Diese Grundstruktur beinhaltet eine „Blaupause" als Grundlage für die weitere Kurskonstruktion. Es wird also lediglich ein didaktisch begründetes Gerüst für den Kurs geliefert, welches die Autorin „zu füllen" hat.

Dabei wurden sechs Input-Variablen gewählt, die die Autorin zu spezifizieren hat, und von denen die Struktur des „vorfabrizierten Interaktionsraumes" abhängig gemacht wird. Es handelt sich um folgende Fragen:

- Inhaltsstruktur: Bauen die einzelnen Lehr-Lerninhalte aufeinander auf/ sind sie unabhängig voneinander?
- Nutzungssituation: Wird das Medium als Nachschlagewerk und/oder als Ersatz für Unterricht eingesetzt?
- Homogenität: Ist die Zielgruppe insbesondere hinsichtlich ihres Vorwissens homogen/inhomgen?
- Vorwissen: Ist das Vorwissen im Durchschnitt eher niedrig/hoch (d.h. sind die grundlegenden Begriffe und Konzepte bekannt)?
- Anlaß: Ist die Teilnahme am Kurs freiwillig/verpflichtend oder Teil eines formalen Lehrgangs?
- Interesse: Ist das Interesse an der Kursthematik hoch/niedrig?

Nach Eingabe der Daten wird nun ein Vorschlag unterbreitet, wobei unterschieden wird zwischen:

- linearer Interaktionsraum,
- offener Interaktionsraum mit Pfadvorgabe oder
- hyperstrukturierter Interaktionsraum (i.e. hohe Vernetzung der Informationseinheiten).

Die Zuordnungskriterien der Eingaben zu diesen Varianten von Interaktionsräumen beschreiben Müller & Schweitzer (1994). Ein hyperstrukturierter Interaktionsraum wird etwa eher vorgeschlagen, wenn die Lehrinhalte wenig stark aufeinander aufbauen, die Benutzung auch als Nachschlagewerk vorge-

sehen ist oder hohes Vorwissen angenommen werden kann. Eine eher sequentielle Organisation in einem linearen Interaktionsraum wird dagegen empfohlen bei stärker aufeinander aufbauenden Lehrinhalten, tendenziell bei niedrigem Vorwissen sowie bei eher verpflichtender Teilnahme. Darüber hinaus liegen auch Kombinationen vor, die für den Einsatz eines computergestützten Lehr-Lernmediums tendenziell ungünstig sind, so daß von einer entsprechenden Implementation abgeraten wird.

Das Programm macht einen *Vorschlag* zum Interaktionsraum, die Autorin kann diesen akzeptieren oder eine andere Variante wählen. Generiert wird nun die Grundstruktur des entsprechenden Interaktionsraumes sowie ein Autorenmenü, das die Kurserweiterungen managt. Parallel zur Anwendung werden dabei strukturelle Daten der Anwendung registriert, um so z.B. die Verschachtelungstiefe zu erfassen. Zukünftige Erweiterungen des CBT-Guides könnten derartige Daten dann auch in der Phase der Entwicklung auswerten und Rückmeldungen geben z.B. über zu tiefe bzw. flache Interaktionsräume.

Worin liegt nun der Wert eines solcher Werkzeugs? Hinsichtlich der vom Programm gemachten Zuordnungen der Input-Variablen zu bestimmten Interaktionsräumen kann die gleiche Kritik vorgebracht werden wie bei ID-Expert und DISCourse, nämlich daß die zugrunde liegenden Zuordnungen keineswegs zwingend bzw. durch deskriptive Forschungsergebnisse belegbar sind. In einem Unterstützungssystem kommt diesen „Präskriptionen" jedoch eine andere *Bedeutung* zu als in den genannten stärker automatisierenden Entwicklungsumgebungen: Sie beinhalten lediglich Empfehlungen für Autorinnen und dienen der didaktischen Konzeptualisierung des Mediums. Ein solches System fördert damit m.E. das konzeptuelle „mediendidaktische Denken" von CBT-Autorinnen, und kann insofern insbesondere für Ausbildungszwecke eingesetzt werden. Für die standardmäßige Anwendung in professionellen Medienproduktionen erweist sich das Raster mit sechs Variablen (das aus theoretischer Sicht bereits als relativ gewagt erscheinen mag) dagegen als zu beschränkt. Letztlich sind es gerade die schriftlich ausgearbeiteten Materialien, die den Aufbau dieser mediendidaktischen Expertise bei Anfängern fördern. Der CBT-Guide dient dann eher der Veranschaulichung und Problematisierung der diskutierten Konzepte in der Ausbildung denn als produktivitätssteigerndes Werkzeug.

Eine andere Entwicklung betrifft *analytische Werkzeuge,* die Anwendungen auf Schwachstellen untersuchen und gezielte Hinweise für Verbesserungen geben. Die Unterstützung bezieht sich also nicht mehr auf die Entwurfsphase des Interaktionsraumes, sondern es werden bereits (teilweise) fertige Interaktionsräume analysiert.

Wagner (1995) prüft die Entwicklung eines Werkzeuges für die Untersuchung von „Interaktionsräumen", die z.B. folgende strukturelle „Schwächen" aufweisen können:

- unausgewogene Äste des Hierarchiebaumes,
- unausgewogene Verschachtelung von Teilen des Hierarchiebaumes,

- „Sackgassen", die nicht angemessen zu anderen Teilen des Hierarchiebaumes zurückführen,
- insgesamt zu „tiefe" Auslegung des Hierarchiebaumes,
- insgesamt zu „breite" Auslegung des Hierarchiebaumes,
- insgesamt zu niedrige/hohe Vernetzung des Interaktionsraumes (im Hinblick auf Thema / Benutzer / Benutzungssituation).

Manche derartiger „Fehler" sind bereits auf Schwächen der Konzeption (im *interaktiven Drehbuch*) zurückzuführen. Insofern wäre es wünschenswert, wenn diese bereits dort identifiziert werden könnten. Dies ist jedoch solange nicht möglich, wie die Erstellung dieser Konzeptionen weiterhin mit „Papier-und-Bleistift" erfolgt. Durch den Mangel an analytischen Werkzeugen werden manche dieser Schwächen überhaupt erst in der Phase der Testung registriert, was einen ungerechtfertigt hohen Modifikationsaufwand nach sich zieht.

Die Identifikation derartiger Fehler in der *Entwicklungsphase* bleibt bislang der menschlichen Prüfung überlassen, was bei komplexen, etwa stark hyperstrukturierten Interaktionsräumen äußerst schwierig ist.

Ein derartiges analytisches Werkzeug dient als Instrument, um mögliche Schwachstellen in der Entwicklungsphase registrieren zu können; es bleibt die Aufgabe der Autorin, über die Angemessenheit derartiger Hinweise zu entscheiden. Eine weitergehende Automatisierung, z.B. im Sinne einer „Korrektur" der Anwendung, erscheint ungerechtfertigt, da es sich hierbei lediglich um Erfahrungs- und Plausbilitätsüberlegungen handelt.

In der Umsetzung eines solchen analytischen Werkzeugs zeigen sich jedoch massive technische Probleme, die daraus resultieren, daß die Vorgehensweise erfahrener Autorinnen so vielschichtig ist, daß ein entsprechendes Werkzeug dies kaum abfangen kann. So wäre auch hier der Einsatzbereich wiederum auf Anfänger und die Ausbildung von Instruktionsdesignern beschränkt.

Im Unterschied zu den möglichst umfassenden Entwicklungsumgebungen adressieren die dargestellten Werkzeuge bewußt nur Teilaspekte des Planungs- und Entwicklungsgeschehens. Es zeigt sich, daß bisherige Werkzeuge vor allem der mediendidaktischen Ausbildung dienen.

6. Ratgebersysteme

Didaktische Ratgebersysteme können Instruktionsdesigner in den verschiedenen Phasen der Planung und Entwicklung unterstützen, z.B. bei der Entscheidungsfindung durch Hinweise zu Medienwahl und Medieneinsatz, vorwiegend in textlicher Form.

Instruktionsdesigner können Ergebnisse dieser didaktischen Ratgebersysteme nutzen und in weiteren Schritten einbeziehen. Eine „Automatisierung"

im eigentlichen Sinne findet dabei nicht statt. Das Ratgebersystem trägt dazu bei, das Wissen über Unterrichtsplanung und -entwicklung zu systematisieren und vermittelbar zu machen. Ähnliche „electronic perfomance support systems" (EPSS) finden wir seit längerem z.B. im medizinischen oder juristischen Bereich.

G-AIDA (Guided Automated Instruktional Design Advisor) von Spector & Muraida (1993) ist eine Toolbook-Anwendung, die den CBT-Entwicklern der US-Air Force Hinweise zur Kursgestaltung geben sollen (s.a. Spector & Song, 1995). Erläutert wird die Lehrtheorie von Gagné *(Events of Instruction);* es folgen Beispiele aus dem militärischen Bereich, wie das schrittweise Vorgehen in Lehrprogrammen angewandt werden kann. Zunächst beinhalten die Erläuterungen des theoretischen Modells lediglich Materialien, die auch in textlicher Form vorliegen. Die Beispiele führen dem CBT-Autor jedoch unmittelbar vor, wie das theoretische Modell in einem computerbasierten Lehrprogramm umgesetzt werden kann. Kulturspezifische Eigenarten sind nicht zu übersehen.

Das Ratgebersystem in der Diplomarbeit von Lechner (1994) behandelt die Problematik der „Navigation" vor allem in stärker vernetzten, hyperstrukturierten Interaktionsräumen. Der Autor muß die Orientierung des Lerners in einem solchen System sicherstellen (Verhinderung von „lost in hyperspace", Aufbau kohärenter kognitiver Schemata etc.). Lechner beschreibt Möglichkeiten globaler und lokaler Navigation und gibt präzise Gestaltungshinweise für Autoren. Die Überlegungen münden in eine CBT-Anwendung, die Autoren bei der Planung und Gestaltung von Navigation und Navigationsinstrumenten in ihren Anwendungen unterstützen sollen, indem übernehmbare Instrumente zur Verfügung gestellt werden.

ConStruct, einen Ratgeber für Instruktionsdesign, entwickelten Lowyck, Elen & van den Branden (1990) (s.a. Elen & Stevens, 1993). Das Programm ist nicht auf computergestützte Medien festgelegt, sondern gibt – auf der Grundlage von zuvor erfragten Informationen über Zielgruppe, Randbedingungen etc. – Hinweise für die Wahl der Vermittlungsform (direkter Unterricht, CBT u.a.), die Gestaltung der jeweiligen Medien und zur Unterrichtsdurchführung. Das Programm beruht auf einer Datenbank über wissenschaftlichen Literaturbefunde zum Instruktionsdesign. Es sind Hinweise gespeichert zu verschiedenen Ausprägungen von Variablen in den Bereichen Unterrichtsinhalte, -ziele, -formen sowie Lernermerkmale. In Abhängigkeit von den Antworten wird ein Text zusammengestellt, der dem Ratsuchenden abschließend präsentiert wird. Der Benutzer kann daraufhin weitere, präzisere Informationen abfragen. Ebenfalls auf verschiedene Unterrichtsformen ist das Ratgebersystem von Flechsig & Haller (1992) angelegt.

Auch für den Bereich der Medienwahl als Teilproblem des Instruktionsdesigns existieren Programme, die vorhandene theoretische Modelle praktisch nutzbar machen. Reiser & Gagnés (1983) Modell der Medienwahl etwa ließ sich noch ohne Computereinsatz anwenden; komplexere Entscheidungsmo-

delle wie Seels "Multiple-Attribute-Utility-Test", als interaktives entscheidungsunterstützendes Instrument, läßt sich dagegen nur mithilfe von Computern anwenden (vgl. Seel et al., 1995).

Bei der Entwicklung von Ratgebersystemen stellt sich grundsätzlich die Frage, wie das System zu „füllen" ist: woher stammt das „didaktische Expertenwissen"? Man kann sich entweder an dem explizit verfügbaren präskriptiven Wissen über die Gestaltung von Lehr-Lernprozessen oder dem zumeist impliziten Wissen von Instruktionsdesignern orientieren.

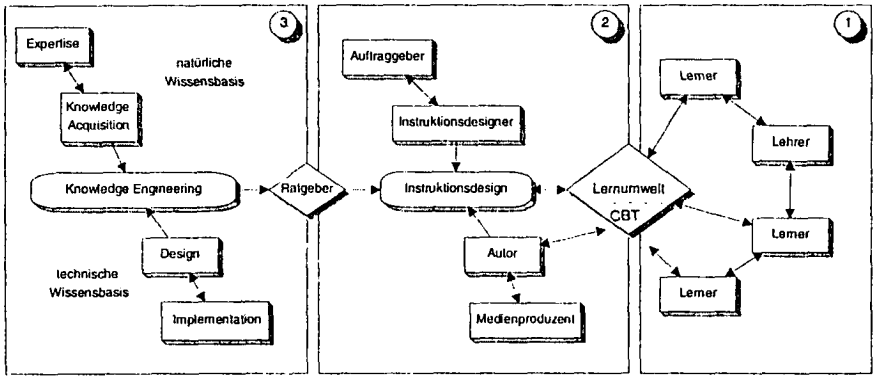
Der erste Weg ist von Lowyck, Elen & van den Branden (1990) eingeschlagen worden, indem sie ein Ratgebersystem entwickelten, in dem vor allem Wissen über die Gestaltung didaktischer Textmaterialien aufgearbeitet und implementiert wurde. Grenzen ergeben sich dort, wo keine wissenschaftlich begründeten bzw. begründbaren Erkenntnisse vorliegen. In bestimmten Bereichen ist dieses abstrakte Wissen grundsätzlich beschränkt, da die Komplexität realer Designentscheidungen einfachen Abbildungen und Abstraktionen entgegenstehen.

Deswegen bietet es sich an, den Entwicklungsprozeß von Instruktionsdesignern zu untersuchen, um diesen ggf. nachzubilden (vgl. Perez & Neiderman, 1992; Latzina & Schott, 1995). Praktisch tätige Instruktionsdesigner haben im Laufe ihrer beruflichen Tätigkeit ein mehr oder weniger komplexes Geflecht deklarativen und prozeduralen Wissens aufgebaut, das es im Sinne eines „Knowledge Engineering" zu explizieren gilt (vgl. Abbildung 2).

Dabei ist die didaktische Expertise zu erfassen ("knowledge acquisition") und zum anderen in eine formalisierbare Darstellung zu transformieren, um diese, etwa in einer Wissensbasis, implementierbar zu machen ("knowledge base design"). Es wird die Annahme zugrunde gelegt, daß das Wissen über Instruktionsdesign als weitgehend generisches, situationsunabhängiges Wissen implementierbar sei, das „aus dem Kopf" eines Experten in einen Computer übertragen wird, und durch das Abrufen von Anderen wieder verfügbar wird.

Die praktische Erfahrung mit der Entwicklung von Expertensystemen hat deutlich gemacht, daß dieses technologische Modell des Explikations- und Implementationsprozesses problematisch ist (vgl. z.B. Hoffman, 1992). Die Konstruktivismus-Diskussion hat schließlich dazu beigetragen, die zugrunde liegende Konzeption von „Wissen" infrage zu stellen (vgl. Brown, Collins & Duguid, 1989, aber auch Dinter & Seel, 1994): Demnach wäre „Wissen" nicht in den Köpfen von Instruktionsdesignern „gespeichert", um von da auf Situationen „angewandt" zu werden. Vielmehr würde dies in konkreten Anforderungszusammenhängen jeweils neu konstruiert als Ergebnis des Zusammenwirkens verschiedener in der Situation wirksamer Komponenten.

Abbildung 2: Didaktisches Knowledge Engineering



Aus dieser Sicht wäre es fragwürdig, ob es möglich bzw. sinnvoll ist, einen generischen Didaktik-Ratgeber zu entwickeln, der – programmatisch – die Situiertheit des Instruktionsdesigns ausblendet.

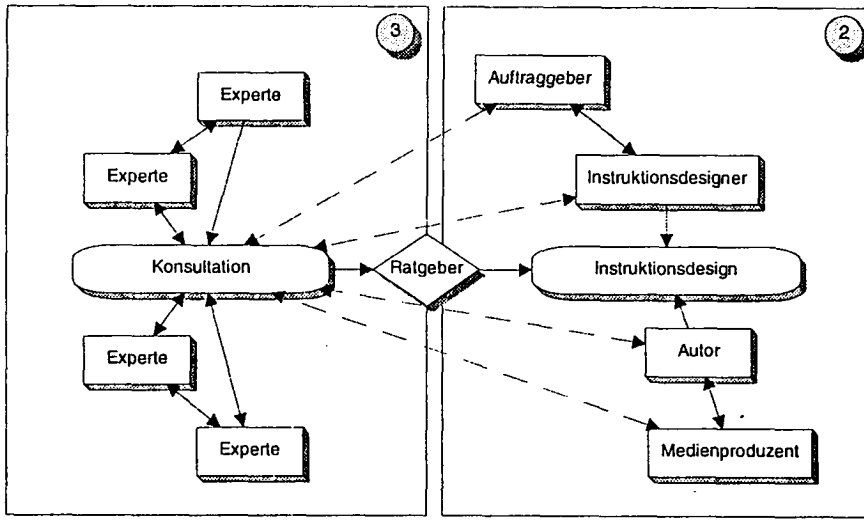
Welche Konsequenzen ergeben sich hieraus für Ratgebersysteme zum Instruktionsdesign? Als Alternative zum „Knowledge Engineering"-Ansatz als Modell zur Entwicklung eines didaktischen Ratgebersystems bietet sich ein *Konsultationssystem* an, wie es im Rahmen von JITOL konzipiert wurde (vgl. Dobson et al., 1993). Es geht davon aus, daß Wissen über Instruktionsdesign in konkreten Projekten konstruiert wird. Dabei kann der Dialog des Instruktionsdesigners mit „Instruktionsdesign-Experten" nützlich sein und bei der Bearbeitung eines konkreten Projektes helfen. Da diese Expertise üblicherweise nicht vor Ort abrufbar ist, können kommunikationstechnische Hilfsmittel genutzt werden, um den Dialog zwischen Instruktionsdesigner und entfernten Experten zu unterstützen.

Relativ schnell haben sich elektronische Konferenzen, wie im Internet, etabliert. Teilnehmer der Konferenz diskutieren Fragen zum Thema Instruktionstechnologie und holen Rat von Anderen ein. Diese Konferenzen vereinen eine immer wechselnde Mischung von Wissenschaftlern und Praktikern mit theoretischen wie praktischen Diskussionen.

Im Unterschied zu einer solchen unstrukturierten elektronischen Konferenz steht bei einem *Konsultationssystem* das Ergebnis der Beratung nicht nur der fragenden Person zur Verfügung, sondern fließt in einen Ratgeber ein und steht damit auch anderen Interessenten zur Verfügung („knowledge reification", vgl. Boder, 1992). Würde man in dieses Ratgebersystem *alle* Dialoge und Hinweise unkontrolliert ohne weitere Strukturierung aufnehmen, entsteht innerhalb kürzester Zeit ein ungeordnetes Chaos, das den Nutzen eines entsprechenden Ratgebers einschränkt (vgl. News-Groups im Internet, s.a. Roschelle & Behrend, 1993; s. a. Goodyear & Steeples, 1992, zur Konzeption der Ausbildung von CBT-Autoren mit Hilfe von Kommunikationstechniken im Rahmen des JITOL-Projektes des europäischen DELTA-Pro-

gramms). Auch die Strukturierung durch einen Moderator kann keine zufriedenstellende Lösung beinhalten, da das Ergebnis zu sehr von der idiosynkratischen Wertung des Moderators abhängt und darüber hinaus der Bewertungs- und Strukturierungsaufwand hoch ist.

Abbildung 3: Entferntes Konsultationssystem für Instruktionsdesign



Es wäre folglich ein Filter zu schaffen, der die Konsultationsergebnisse festhält und in einer geeigneten Form dem Ratgebersystem zur Verfügung stellt. Dies bedeutet insbesondere für die Dialogführung zwischen den Partnern des Konsultationsprozeß, daß ein Formalismus einzuhalten ist, der die automatische Rückführung in das Ratgebersystem möglich macht (vgl. Abbildung 3).

Ein frühes computergestütztes Ratgebersystem ist MIT's „On-Line-Consultant" (Coppeto et al., 1989). Es können Anfragen (technische Probleme) an das System gestellt werden, welche automatisch an den „richtigen" verfügbaren Berater weitergeleitet werden. In der Folge wurde das System erweitert: *AnswerGarden* beinhaltet eine Datenbasis mit Antworten zu häufig gestellten Fragen, die per Computer abfragbar sind. Liegt in der Datenbasis nun (noch) keine Antwort für das eigene Problem vor, wird diese an einen Berater weitergeleitet. Seine Antwort auf die Frage wird gleichzeitig in das System eingespeist und führt damit dazu, daß das System mit zunehmender Nutzung wächst (vgl. Ackerman & Malone, 1990).

7. Schluß

Seit längerem sind die Probleme von Autorensystemen bekannt, ebenso lange wird an Alternativen gearbeitet, und immer schwanken die Ansätze zwischen der Hoffnung einer weitgehenden Automatisierung und Skepsis angesichts der Probleme, die sich bei der praktischen Umsetzung ergeben.

Eine realistische Sicht muß konstatieren, daß die Ziele Produktivitäts- und Qualitätssteigerung zur Zeit auseinanderlaufen: Ansätze, die die Produktivität prinzipiell verbessern könnten (Entwicklungsumgebungen), verhindern eher die Erhöhung didaktischer Qualität. Diese wiederum ließe sich mit Ratgeber- und Konferenzsystemen erreichen, was jedoch keine unmittelbare Produktivitätssteigerung mit sich bringt.

Bisherige Ansätze haben die Bedürfnisse potentieller Benutzer nicht deutlich genug berücksichtigt und differenziert. Im wesentlichen zu trennen bleiben Bemühungen, die letztlich der Ausbildung von Instruktionsdesignern dienen, von Anforderungen, die sich aus der professionellen Entwicklung von Lehr-Lernmedien ergeben.

Für die Ausbildung sind Unterstützungssysteme notwendig, die dazu beitragen, daß mediendidaktische Kompetenz aufgebaut wird. Stark automatisierende Systeme „verhindern" zwar die Auseinandersetzung mit Fragen der Sequenzierung von Lehrangeboten, gleichwohl bieten sie eine Lernumgebung für Autoren, indem sie zur systematischen Aufbereitung von Lehrinhalten anleitet. Gerade dies wird von Autorensystemen vernachlässigt.

Erfahrene Instruktionsdesigner in der Medienproduktion akzeptieren andererseits nur Werkzeuge, die an ihrer bisherigen Arbeitsweise ansetzen und diese in keiner Weise einschränken. Versuche alternative Vorgehensweise zu etablieren scheitern hieran. Produktive wie analytische Design- und Entwicklungswerkzeuge schlagen fehl, weil sie nicht annähernd die Komplexität erfahrener Instruktionsdesigner nachbilden können. Ratgebersysteme könnten für begrenzte Fragestellungen interessant sein; es bleiben Konsultationssyteme als Forum für Experten.

Die Konfrontation mit grundlegenden und teilweise unüberwindbaren Problemen, denen sich in den 60er Jahren bereits die europäischen Bildungskybernetiker ausgesetzt sahen, wird möglicherweise der Idee einer „weak automation" Vorschub leisten. Damit sind Systeme gemeint, die nicht beanspruchen, didaktische Medien selbsttätig generieren zu können, sondern sich darauf beschränken, Autoren bei der Entwicklung solcher Medien zu unterstützen. Vielleicht müssen wir zurückhaltender sein in Wunschvorstellungen über die Machbarkeit maschineller Lösungen; vielleicht sind es dann eher die „kleinen", wenngleich unspektakulären Lösungen, die die Entwickler in der Zukunft unterstützen.

Literatur

Ackermann, M. & Malone, T. (1990). AnswerGarden: a tool for growing organisational memory. Proceedings of the ACM Conference on Office Information Systems. S. 31-39.

Baker, M. (1994). Adapting instructional design methods to intelligent multimedia authoring systems. In R. Tennyson (Hg.) *Automating Instructional Design, Development, and Delivery. (NATO ASI Serie F. 119).* (S. 197-223) Berlin: Springer.

Boder, A. (1992). The process of knowledge reification in human-human interaction. *Journal of computer assisted learning, 8,* 177-182.

Boder, A. & Gardiol Gutierrez, Ch. (1992). *Multimedial Lernsysteme in Europa herstellen. Ergebnisse des europäischen Projektes Start-up, Wissenschaftspolitik, Beiheft 54.*

Brown, J.S., Collins, A. & Duguid, P. (1989). Situated cognition and the culture of learning. *Educational researcher, 18,* 32-42.

Bunderson, C. V. (1974). Design strategy for learner-controlled courseware. In K. Brunnstein, K. Haefner & W. Händler (Hg.) *Rechner-Gestützter Unterricht (RGU '74 Fachtagung).* (S. 308-322) Berlin: Springer.

Chiocariello, A., Innocenti, C., Persico, D., Sarti, L., Viarengo, V. & van Marcke, K. (1993). New conceptual scheme description for DBLM. *DELTA deliverable, 2008/27.*

Cleary, A., Mayes, T. & Packham, D. (1976). *Educational technology: Implications for early and special education.* New York: Wiley & Sons.

Coppeto, T., Anderson, B., Geer, D. & Treese, G. (1989). LOC: an On-Line Consulting System for UNIX. Proceedings of the Usenix Summer 1989 Conference. S. 83-94.

Dinter, F.R. & Seel, N.M. (1994). What does it mean to be a constructivist in I.D.? An epistemiological reconsideration. In J. Lowyck & J. Elen (Hg.) *Modelling I.D.research. Proceedings of the first Workshop of the Special Interest Group on Instructional Design of EARLI, Leuven.* (S.49-66).

Dobson, M., Rada, R., Chen, C., Michailidis, A. & Ulloa, A. (1993). Towards a consolidated model for a collobarative courseware authoring system. *Journal of computer assisted learning, 9,* 34-50.

Elen, J. & Stevens, W. (1993). A coach for instructional design decision making (Paper presented at the Instructional Design SIG-symposium, 5th EARLI-conference, Aix-en-Provence).

Elsom-Cook, M. & O'Malley, C.: ECAL. Bridging the gap between CAL and ITS. CITE Report #67, London: The Open University 1989.

Eyferth, K. (1974). *Computer im Unterricht: Formen, Erfolge & Grenzen einer Lerntechnologie in der Schule.* Stuttgart: Ernst Klett Verlag.

Fischer, K. & Löthe, H. (1975). PLATO IV – ein umfassendes CUU-System. In K. Boeckmann & U. Lehnert (Hg.) *Fortschritte und Ergebnisse der Unterrichtstechnologie Bd. 3.* (S. 228-239) Paderborn/Hannover: Schöningh/Schroedel.

Frank, H. (1966). Ansätze zum algorithmischen Lehralgorithmieren. In H. Frank (Hg.) *Lehrmaschinen in kybernetischer und pädagogischer Sicht, Bd. 4.* (S. 70-112) Stuttgart: Ernst Klett Verlag.

Frank, H. (1969). Prinzipien der objektivierten Formaldidaktik 'Alskindi'. *Grundlagenstudien aus Kybernetik und Geisteswissenschaft, 10,* S. 23-28.

Frank, H. & Graf, K.D. (1967). ALZUDI – Beispiel einer formalisierten Didaktik. *Grundlagenstudien aus Kybernetik und Geisteswissenschaft, 8,* 27-34.

Goodyear, P. & Steeples, C. (1992). IT based open learning: tasks and tools. *Journal of computer assisted learning, 8,* 163-176.

Goodyear, P. (1994). Foundations for courseware engineering. In R. Tennyson (Hg.) *Automating Instructional Design, Development, and Delivery. (NATO ASI Serie F. 119).* Berlin: Springer.

Gros, B. (1994). The automation of instructional design: How to find the appropriate direction. In J. Lowyck & J. Elen (Hg.) *Modelling I.D.research. Proceedings of the first Workshop of the Special Interest Group on Instructional Design of EARLI, Leuven.* (S.125-140).

Gustafson, K. & Reeves, T. (1990). IDIOM: A Plattform for a course development expert system. *Educational Technology, 31,* 19-25.

Hawkridge, D. (1988). The state of computer based training after 25 jears. In J. Finkelstein & T. Bernold (Hg.) *Computer Assisted Approaches to Training. New York: North-Holland.*

Hoffman, R.R. (Hg.) (1992). The psychology of expertise: Cognitive research and empirical AI. New York: Springer.

Issing, L.J. (Hg.) (1967). Der Programmierte Unterricht in den USA heute. Weinheim: Beltz.

Jonassen, D.H. (1991). Objectivism versus Constructivism: Do we need a new philosophical paradigm? *Educational Technology: Research & Development, 39 (3),* 5-14.

Jones, M. & Wipond, K. (1991). Intelligent environments for curriculum and course development. In P. Goodyear (Hg.) *Teaching knowledge and intelligent tutoring.* (S. 379-395) Norwood, NJ: Ablex.

Kerres, M. (1991). Computer-Based Training: Ein deutsches Problem? *Lernfeld Betrieb, 4,* S.44-47.

Kerres, M. (1993). Software-Engineering für multimediale Teachware. In C. Seidel (Hg.) *Computer Based Training.* (S. 87-102) Stuttgart: Verlag für Angewandte Psychologie.

Kerres, M. (1995). Integrating 'computer assisted learning' into the organisational context as an instructional design task. *Journal of computer assisted learning* (im Druck).

König, E. & Riedel, H. (1979). *Unterrichtsplanung I. Konstruktionsgrundlagen und -kriterien.* (2. Aufl.) Weinheim: Beltz.

Latzina, M. & Schott, F. (1995). Psychological processes of planning in instructional design teams: Some implications for automating instructional design. In R. Tennyson & A. Barron (Hg.) *Automating instructional design: Computer-based development and delivery tools.* (S.131-148) New York: Springer.

Lechner, M. (1994). *Ergonomische Navigation in computerbasierten Informationssystemen. Bestandsaufnahme und Entwicklung eines CBT-Programms.* Diplomarbeit im Fachbereich Medieninformatik der Fachhochschule Furtwangen.

Lowyck, J. & Elen, J. (1991). Wandel in der theoretischen Fundierung des Instruktionsdesigns. *Unterrichtswissenschaft, 19,* 218-237.

Lowyck, J., Elen, J. & van den Branden, J. (1990). ConStruct: A computercoach for the development of high quality study materials (paper presented at 7th Intl. Conference on Technology for Education).

Ludwig, E.H. (1965). *Die Technik zur Herstellung von Lehrprogrammen für die programmierte Unterweisung.* Ratingen: Henn.

Mandl, H., Hron, A. & Tergan, S.O. (1990). *Computer-Based Systems for Open Learning. State of the Art (DELTA Deliverable, PRECISE 7065).*

McIntyre, A. (1993). A computational framework for representing authors' courseware models. Journal of computer assisted learning, 9, 244-261.

Merrill, M. D. (1983). Applying component display theory to the design of courseware. In C. Reigeluth (Hg.) *Instructional Design Theories.* (S. 61-95) Hillsdale, NJ: Lawrence Erlbaum Associates.

Merrill, M.D. (1987). The new Component Design Theory: Instructional design for courseware authoring. *Instructional Science, 16,* 19-34.

Merrill, M.D., Li, Z. & Jones, M.K. (1991). Instructional transaction theory: An introduction. *Educational Technology, 31,* 7-12.

Müller, R. & Schweiter, A. (1994). *Ratgebersystem für CBT-Autoren.* Diplomarbeit im Fachbereich Medieninformatik der Fachhochschule Furtwangen.

Ohlsson, S. (1993). Impact of cognitive theory on the practice of courseware authoring. *Journal of computer assisted learning, 9,* 194-221.

Pagliaro, L.A. (1983). The history and development of CAI: 1926-1981, an overview. *Alberta journal of educational research, 29,* 75-84.

Perez, R. & Neiderman, E.C. (1992). Modelling the expert training developer. In R. Seidel & P. Chatelier (Hg.) *Advanced training technologies applied to training design. (S. 8-15) Hillsdale, NJ: Lawrence Erlbaum Associates.*

Perez, R. & Seidel, R. (1990). Using artificial intelligence in education: Computer-based tools for instructional development. Educational Technology, 31, 46-50.

Posner, G.J. & Rudnitzky, A.N. (1986). *Course design: A guide to curriculum development for teachers.* New York: Longman.

Ranker, R. & Doucet, R. (1990). SOCRATES: A computer-based lesson development advisor. *Educational Technology, 31,* 46-50.

Reimann, P. (1993). Commentary to MyIntyre (1993). *Journal of computer assisted learning, 9,* 262.

Reiser, R.A. & Gagné, R.M. (1983). *Selecting media for instruction.* Englewood Cliffs, NJ: Educational Technology Publications.

Roschelle, J. & Behrend, S.D. (1993). The construction of shared knowledge in collaborative problem solving. In C. O'Malley (Hg.) *Computer supported collaborative learning.* Berlin: Springer.

Russel, D.M., Morgan, T.P. & Jordan, D.S. (1988). The instructional design environment. In J. Psotka, L. Massey & S. Mutter (Hg.) *Intelligent tutoring systems: Lessons learned.* Hillsdale, NJ: Lawrence Erlbaum Associates.

Schiffman, S.S. (1991). Instructional systems design. Five views of the field. In G. Anglin (Hg.) *Instructional technology: Past, present, and future.* Englewood, CO: Libraries Unlimited.

Seel, N. M. (1981). *Lernaufgaben und Lerprozesse.* Stuttgart: Kohlhammer.

Seel, N. M. (1991). Lernumgebungen und institutionell-organisatorische Bedingungen des Instruktionsdesigns. *Unterrichtswissenschaft, 19 (4),* 350-364.

Seel, N.M. , Eichenwald, L.D. & Penterman, N.F.N. (1995). Automating decision support in instructional system development: The case of delivery systems. In R. Tennyson & A. Barron (Hg.) *Automating instructional design: Computer-based development and delivery tools.*(S.177-215) New York: Springer.

Shuell, T.J. (1988). The role of the student in learning from instruction. *Contemporary educational psychology, 13,* 276-295.

Spector, M. & Muraida, D. (1990). *Designing and developing an advanced instructional design advisor. Brooks Air Force Base, Texas: Armstrong Laboratories.*

Spector, J.M. & Song, D. (1995). Automating instructional design advising. In R. Tennyson & A. Barron (Hg.) *Automating instructional design: Computer-based development and delivery tools.* (S.377-402) New York: Springer.

Steinberg, E.R. (1989). Cognition and learner control: A literature review. *Journal of computer-based instruction, 16,* 117-121.

Tait, K. (1992). The description of subject matter and instructional methods for computer-based learning. In S. Dijkstra, H. Krammer & J. van Merriënboer (Hg.) *Instructional models in computer-based learning environments (NATO ASI Serie F, Vol. 104).* (S.127-141) Berlin: Springer.

Tennyson, R.D. , Spector, J.M. & Muraida, D.J. (1994). Automation and instructional system development. In R. Tennyson (Hg.) *Automating Instructional Design, Development, and Delivery. (NATO ASI Serie F. 119).* (S. 1-6) Berlin: Springer.

Ulloa, A. (1991). Computer aided multimedia courseware engineering (CAMCE final report). *DELTA deliverable, 1015.*

Ulloa, A. (1994) (Hg.). System Architecture, INDIOS. DELTA Deliverable 1, Project D2043.

van Marke, K. (1992). A generic task model for instruction. In S. Dijkstra, H Krammer & J van Merriënboer (Hg.) *Instructional models in computer-based learning environments (NATO ASI Serie F, Vol. 104).* (S. 171-193) Berlin: Springer.

Wagner, M. (1995). Untersuchung von Interaktionsräumen in CBT-Anwendungen. Diplomarbeit im Fachbereich Digitale Medien der Fachhochschule Furtwangen.

Wasson, B. J. (1992). PEPE: A computational framework for a content planner. In S. Dijkstra, H. Krammer & J. van Merriënboer (Hg.) *Instructional models in computer-based learning environments (NATO ASI Serie F, Vol. 104).* Berlin: Springer.

Wasson, Barbara (1994). Automating the development of intelligent learning environments: A perspective on implementation issues. In R. Tennyson (Hg.) *Automating Instructional Design, Development, and Delivery. (NATO ASI Serie F. 119).* (S.163-190) Berlin: Springer.

Wenger, E. (1987). *Artificial intelligence and tutoring systems.* Los Altos, CA: Morgan Kaufmann.

Wittmann, N. (1995). Die Rolle des Autors im automatisierten Instruktionsdesign. Diplomarbeit im Fachbereich Medieninformatik der Fachhochschule Furtwangen.

Wulfeck, W.H. , Dickieson, J.L. , Apple, J. & Vogt, L.J. (1993). The automation of curriculum development using the Authoring Instructional Materials (AIM). *Instructional Science, 21,* 255-267.

Zifreund, W. (1965). Ein strukturanalytisches Diagramm als Hilfsmittel bei der Herstellung und kritischen Analyse von Lehrprogrammen. In H. Frank (Hg.) *Lehrmaschinen in kybernetischer und pädagogischer Sicht, Bd. 3.* (S. 114-135) Stuttgart: Ernst Klett Verlag.

Anschrift des Verfassers:
Prof. Dr. Michael Kerres, Dipl.Psych,
Fachhochschule Furtwangen (Schwarzwald), Fachbereich Digitale Medien,
D-78113 Furtwangen, Telefon (0 77 23) 9 20 56 - 3 / - 0, Fax (0 77 23) 92 06 10
eMail: kerres@fh-furtwangen.de
http://www.mi-Lab.fh-furtwangen.de