Zeitschrift: Bulletin des Schweizerischen Elektrotechnischen Vereins, des Verbandes Schweizerischer Elektrizitätsunternehmen = Bulletin de l'Association suisse des électriciens, de l'Association des entreprises électriques suisses

Herausgeber: Schweizerischer Elektrotechnischer Verein ; Verband Schweizerischer Elektrizitätsunternehmen

Band: 78 (1987)

Heft: 1

Artikel: Arbeitsplatzrechner im technisch-naturwissenschaftlichen Hochschulunterricht

Autor: Fischlin, A. / Schaufelberger, W.

DOI: https://doi.org/10.5169/seals-903795

Arbeitsplatzrechner im technisch-naturwissenschaftlichen Hochschulunterricht

A. Fischlin und W. Schaufelberger

Computer spielen im Hochschulunterricht eine immer grössere Rolle. In der vorliegenden Arbeit werden einige grundsätzliche Fragen zur Zielsetzung und zur Gestaltung von Unterrichtsprogrammen diskutiert. Dabei wird insbesondere auf die Autorensysteme zur Gestaltung von Lernprogrammen eingegangen. Einige Beispiele ergänzen die Diskussion.

Les ordinateurs jouent un rôle de plus en plus important dans l'enseignement universitaire. Quelques questions de base concernant les buts et la conception des logiciels d'enseignement sont discutés dans ce travail. Les langages et systèmes d'auteurs qui jouent un rôle central pour le développement de logiciels sont traités en quelque détail. Quelques exemples sont ajoutés pour compléter la discussion théorique.

Adresse der Autoren

Dr. *A. Fischlin* und Prof. Dr. *W. Schaufelberger*, Projekt-Zentrum IDA, ETH-Zentrum, 8092 Zürich.

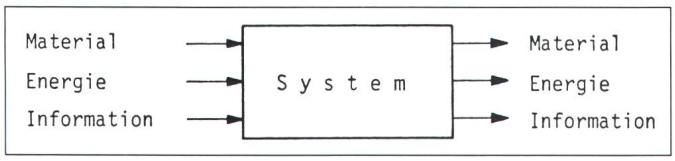
1. Einleitung

Als zu Beginn der 70er Jahre die ersten Mikroprozessoren entwickelt wurden, hat wohl kaum jemand geahnt, welch weitgreifende Auswirkungen diese Technologie dereinst auf die industrielle Produktion, die Verwaltung, den Dienstleistungssektor, die Forschung und Entwicklung sowie den Unterricht haben könnten. Dies geschah just zu einem Zeitpunkt, als die euphorischen Erwartungen, die in den 60er Jahren von einigen Didaktikern an den programmierten Unterricht gestellt worden waren, ersten Ernüchterungen Platz machten [1]. Programmierter Unterricht war nicht nur teuer in der Herstellung, sondern verschlang zudem unverhältnismässige Summen im täglichen Betrieb, d.h. Herstellungs- wie Betriebskosten für Software und Hardware waren im Vergleich zum erwarteten, jedoch ausgebliebenen Nutzen eines gesteigerten Lernerfolges zu hoch, als dass noch grössere Anstrengungen zur Weiterentwicklung des Konzeptes des programmierten Unterrichts sinnvoll schienen.

Heute findet man eine neue Lage vor. Zum einen können die Konsequenzen der Erfindung des Mikroprozessors tagtäglich festgestellt werden, und es zeichnet sich ein Bild ab, das uns ahnen lässt, dass wir erst am Anfang einer Entwicklung stehen, die nicht bloss die Berufswelt, sondern auch den Schulalltag entscheidend umwälzen wird. Dies gilt insbesondere und in zunehmendem Ausmass für den schweizerischen Hochschulbereich, wo sich ein stürmisches Wachstum des Bedarfs nach Arbeitsplatzrechnern anzeigt. Im weiteren sind in der Zwischenzeit eine Vielzahl von neuen Konzepten mit neuen Möglichkeiten für den Computereinsatz entstanden. Nicht mehr bloss numerisches Rechnen oder die Lenkungs- und Überwachungsfunktion des Computers beim programmierten Unterricht stehen heute im Vordergrund, sondern vielmehr professionelle, kostensparende Arbeitsplatzrechner, die auch Kaderleute bei ihrer täglichen Informationsverarbeitung, z.B. beim Ablegen und Wiederauffinden von Notizen, bei der Bearbeitung von Verwaltungsdokumenten, bei der Finanzkalkulation, bei der Terminplanung und beim Verschicken von elektronischer Post, unterstützen [2]. Schon alleine dadurch sind Neuentwicklungen im Bereich der Hard- und Software in Gang gekommen, die die Einsatzmöglichkeiten von Arbeitsplatzrechnern im Unterricht entscheidend verändert haben.

Der Hochschullehrer sieht sich heute konfrontiert mit vielen Einsatzmöglichkeiten von Arbeitsplatzrechnern. Wie soll er wählen? Soll er mit diesen neuen, viel preisgünstigeren Mitteln, z.B. den Personal Computern, den programmierten Unterricht wiederum einführen? Soll der Student an Geräten und Standardsoftware derart ausgebildet werden, dass er das an seinem zukünftigen Arbeitsplatz stehende System sofort gewinnbringend einzusetzen vermag? Ist ein solches Lernziel in Anbetracht der sich äusserst schnell wandelnden Computertechnologie überhaupt vertretbar, und wenn ja, in welchem Masse? Soll er den Computer selbst zum Unterrichtsgegenstand machen, oder soll er ihn vielmehr als neues Medium zur Vermittlung von herkömmlichem Stoff einsetzen? Oder lassen sich mit den modernen Arbeitsplatzrechnern sogar gänzlich neue Unterrichtskonzepte, deren genaue Form zurzeit noch niemand exakt abzuschätzen vermag, entwickeln und einführen?

Bevor auf diese Fragen Antworten gesucht werden, soll dargestellt werden, was den Hochschulunterricht im technischen und naturwissenschaftlichen Unterricht grundsätzlich kenn-



Figur 1
Natürliches oder künstliches (technisches) System

zeichnet und wie sich heute die Situation im einzelnen darbietet. Ingenieure und Naturwissenschafter befassen sich mit Systemen, die Material, Energie und Informationen (Fig. 1) verarbeiten. Sie bestehen dabei aus Teilsystemen und Komponenten, die jeweils wieder Systeme sind und damit eine ganze Betrachtungshierarchie erfordern.

Mey [3] beschreibt die allgemeinen, zeitlichen Wandlungen in der Systembehandlung: Während beim Material der Übergang von der Statik zur Dynamik und bei der Energie derjenige von stationären zu mobilen Quellen bereits weitgehend vollzogen wurde, steht die Informationsverarbeitung mitten im Übergang von einer zentralen zu einer dezentralen Technologie.

Die heutigen Anforderungen, die an Technik und Naturwissenschaften gestellt werden, verlangen, dass sich alle Disziplinen zunehmend der Untersuchung komplexer Systeme zuwenden. Man denke etwa an Verkehrs-, Fertigungs-, Energieversorgungs-, Kommunikations-, Computer- und Ökosysteme. Im beruflichen Alltag ist in Anbetracht dieser Situation eine technisch-naturwissenschaftliche Tätigkeit ohne Computer kaum mehr möglich. Die Einsatzarten reichen von relativ einfacher wissenschaftlicher Textverarbeitung mit Grafik und Formeln sowie Datenverwaltung bis hin zu komplexen Optimierungen und Aufgaben der Prozesssteuerung und Laborautomatisierung. So ist z.B. in vielen Bereichen die Simulationstechnik zur Untersuchung von mathematischen Modellen gleichwertig neben die Theorie und das Experiment getreten.

Diese Ausbreitung der Computeranwendungen in der beruflichen Praxis hat ihr Gegenstück im Hochschulunterricht bis heute erst teilweise gefunden. Man stellt hier unschwer eine sehr grosse Streuung zwischen Fächern fest, die nur mit dem Computer unterrichtet werden können (computerabhängiger Unterricht), wie z.B. Computer Aided Design im Maschinenbau und der Entwurf hochintegrierter Schaltungen (VLSI-Design) in der Elektronik, sowie mehr traditionelle Unterrichtsgebiete. Während auf der einen Seite der Einsatz des Computers eine Selbstverständlichkeit ist, wird auf der anderen nur ausnahmsweise davon Gebrauch gemacht.

In diesem Beitrag wird versucht zu zeigen, wie der Computer sinnvoll in den Unterricht integriert werden kann, damit sich u.a. diese grossen Unterschiede, die nach unserer Meinung nicht gerechtfertigt sind, abbauen lassen. Einige Probleme, die sich beim Einsatz von Arbeitsplatzrechnern im Hochschulunterricht zwangsläufig ergeben, und Ansätze zu deren Lösungen werden geschildert. Diese Thematik wird insbesondere im Hinblick auf die neuartige Verwendung von Arbeitsplatzrechnern zum Studium von technischen und naturwissenschaftlichen Systemen und ihrer Dynamik abgehandelt. Gleichzeitig möchten wir zeigen, dass sich schon heutige Personal Computer und fortgeschrittene Arbeitsplatzrechner ausgezeichnet im Unterricht einsetzen lassen.

2. Arten des Rechnereinsatzes und Unterrichtsziele

Es ist schwierig, allgemein akzeptierbare Ziele für den Einsatz eines Rechners im Unterricht zu finden. Die naheliegende Formulierung, der Computereinsatz solle den Unterricht verbessern, ist nicht aussagekräftig und kaum überprüfbar. Erschwerend kommt dazu, dass Computer im Unterricht auf vielfältigste Weise verwendet werden können. In diesem Zusammenhang ist insbesondere zu unterscheiden zwischen dem Unterricht, der das Rechnersystem selbst, hier aufgefasst als eine funktionale Einheit von Soft- und Hardware, zum Unterrichtsgegenstand macht, d.h. dem eigentlichen Informatikunterricht, und dem Fachunterricht in allen übrigen Ingenieur- und Naturwissenschaften. Zudem ist im zweiten Fall auch noch zu unterscheiden zwischen dem computerabhängigen und dem «lediglich» computerunterstützten Fachunterricht. Unterricht z.B. über komplexe dynamische Systeme ist ohne Simulationen und die damit notwendig gewordene entsprechende Rechenleistung, die heute von moderneren Arbeitsplatzrechnern der höheren Leistungsklasse (ab 1 MIPS[1]) schon zufriedenstellend erbracht werden kann, undenkbar. Der Fachunterricht, der weder auf extensives «number crunching» noch die eher seltene Verwaltung riesiger Datenmengen angewiesen ist, setzt Rechner vor allem als neues didaktisches Medium ein. Von diesem wird erwartet, dass es dem Dozenten gestattet, den Stoff wirkungsvoller an den Studierenden heranzubringen.

Für den Hochschulunterricht stellt sich die Frage nach dem Verhältnis dieser verschiedenen Unterrichtsarten. Es ergibt sich einerseits die Forderung nach allgemeinen Informatikgrundkenntnissen, ungeachtet der Fachrichtung. Insbesondere hat die allgemeine Verwendbarkeit der Arbeitsplatzrechner als Werkzeug im ganzen technisch-wissenschaftlichen Tätigkeitsfeld, von der Textverarbeitung bis zur Informationsbeschaffung, es notwendig gemacht, dass jeder Hochschulabsolvent in diesem Bereich über eingehende Informatikkenntnisse verfügt. Andererseits sind auch fachspezifisch bedingte, starke Unterschiede in der Gewichtung der Anforderungen an die übrigen Unterrichtsarten festzustellen. Nach *Zehnder* [4] werden Naturwissenschafter und Ingenieure als «Fachleute mit Informatikkenntnissen» betrachtet. Ohne Berufsinformatiker zu sein, müssen sie über ausreichende Kenntnisse über den zweckmässigen Einsatz der Informationstechnik in ihrem Fachbereich verfügen. Bei diesen Fachleuten würden wir etwa die folgenden Kenntnisse und Fähigkeiten erwarten:

- Übersicht über den zweckmässigen Einsatz von Standardprogrammen (Textverarbeitung, Tabellenkalkulation, Datenbanken, integrierte Software).
- Fähigkeit, kleinere Programme (einige hundert Zeilen Code) in einer höheren Sprache wie Pascal oder Modula-2 selbst zu erstellen. Kenntnis der wesentlichsten Algorithmen und Datenstrukturen.
- Kenntnis der Grundlagen der Betriebssysteme sowie Erfahrung im Umgang mit einem Betriebssystem (z.B. UNIX).
- Kenntnis des grundsätzlichen Aufbaus des Computers, seiner Arbeitsweise und Grenzen (konzises Benutzermodell).

[1] Million Instructions per Second

Zu diesen Minimalforderungen, denen eigentlich jeder Hochschulabsolvent genügen muss, kommen fachspezifische Anforderungen, die sehr unterschiedlich ausfallen können. Als Beispiel soll hier ein Automatisierungsingenieur betrachtet werden. Von diesem wird man heute etwa die folgenden Computerkenntnisse erwarten:

- Simulationstechnik und -sprachen [5] sowie Optimierungsverfahren für den Einsatz bei der Analyse und Synthese technischer Systeme.
- Computer Aided Control System Design [6] zum Entwurf komplexer Regelungen.
- Methoden zur Echtzeitprogrammierung und damit zur Realisierung von «eingebetteten Systemen», d.h. von technischen Systemen, in denen Computer als Bausteine eine wesentliche Rolle spielen [7].
- Lokale Netzwerke, Betriebsinformationssysteme (Polke 1985 [8]) bis hin zu wissensbasierten Systemen [9].
- Software-Engineering, Techniken für das Erstellen grösserer Programme [10; 11].

Schon allein dieses Beispiel zeigt deutlich, dass alle drei Unterrichtsklassen, der eigentliche Informatikunterricht (z.B. unabdinglich für die Echtzeitprogrammierung), der rechnerabhängige Unterricht (z.B. Entwurf und Simulation von Regelsystemen), wie der Unterricht unterstützt durch das Medium Computer (in jedem Fachbereich), alle vonnöten sind und einander zu ergänzen haben. Nur ein gründlich durchdachter Studienplan garantiert, dass diese Kenntnisse und Fähigkeiten am Schluss des Studiums auch tatsächlich vorhanden sind. Der Unterricht in Informatik muss dazu mit dem fachtechnischen Unterricht gut abgestimmt sein.

Wie im folgenden gezeigt wird, wird mit dem Computer besonders die *individuelle* Arbeit der Studierenden unterstützt. Dies führt dazu, dass sich die vorgestellten Lösungen auch ausgezeichnet für die Weiterbildung des in der Berufspraxis stehenden Ingenieurs und Wissenschafters eignen.

3. Lösungsansätze

Lösungsansätze haben sich an der Tradition des Hochschulunterrichts zu orientieren. Nach *Fischer* [12] unterscheidet man etwa die folgenden Unterrichtsmethoden: dozentenorientiert, studentenorientiert, projektorientiert, exemplarisch oder umfassend; sowie die folgenden Unterrichtsformen: individualisierter Unterricht, Gruppenunterricht, Frontalunterricht. Dazu kommen zeitliche Randbedingungen der Studienpläne durch Einteilung der Lehrveranstaltungen in Vorlesung, Übung, Kolloquium, Seminar, Praktikum, Studien- und Diplomarbeit, und damit in Veranstaltungen, die von einer Stunde bis zu mehreren hundert Stunden Dauer reichen.

Der Computer kann als allgemeines Werkzeug grundsätzlich in allen Lehrveranstaltungen eingesetzt werden. Betrachtet man jedoch die entsprechenden Häufigkeiten, stellt man ein klares Übergewicht der Anwendung beim individualisierten Unterricht und beim Kleinstgruppenunterricht (2 Studierende pro Gruppe) fest. Daneben wird vor allem noch die Frontalvorlesung durch Computervorführungen unterstützt.

Für den Dozenten stellt sich nun die Frage, woher er die benötigten Arbeitsplatzrechnersysteme bekommen kann. Was die Hardware und die damit verknüpfte Systemsoftware betrifft, so lässt sich diese Frage recht einfach beantworten. Die Systeme sollten zwecks optimaler Kompatibilität und minimalen Wartungsaufwands innerhalb der Hochschule in beschränkter Typenzahl betrieben werden (Flottenpolitik), und sie sollten möglichst weit verbreitet oder zumindest technologisch allen anderen Systemen stark überlegen sein. Insbesondere entscheidend ist eine durch die Hard- wie Softwarearchitektur entsprechend unterstützte Grafikfähigkeit, z.B. in Form eines hochauflösenden Bitmap-Displays mit einem dazugehörigen Zeigegerät, z.B. einer Maus.

Bezüglich Applikationssoftware liegen die Verhältnisse schon etwas komplizierter. Der Unterricht kann sowohl mit kommerziell erhältlichen Programmpaketen als auch mit eigens für spezielle Veranstaltungen erstellten Lernprogrammen durchgeführt werden. Die ersteren haben den Vorteil, dass der Student ein Werkzeug kennenlernt, das er unter Umständen später auch in seiner beruflichen Tätigkeit einsetzen kann. Der Nachteil liegt darin, dass diese Programme oft komplex sind und damit eine entsprechende Einarbeitungszeit erfordern. Dieser Aufwand lohnt sich meist bloss bei langlebiger Standardsoftware. Spezielle Programme können besser an die Unterrichtssituation angepasst werden, sind aber in der Erstellung aufwendig. Auf Autorenumgebungen, die das Erstellen von Unterrichtsprogrammen vereinfachen, wird im nächsten Abschnitt eingegangen.

Lernprogramme können nach ganz verschiedenen Gesichtspunkten und Unterrichtszielen entworfen werden. *Forte* [13] unterscheidet etwa:

- Test und Prüfungsprogramme mit Auswahlantworten (programmierter Unterricht).
- Drill- und Trainingsprogramme zur Erlangung bestimmter Fähigkeiten (programmierter Unterricht),
- Abfrageprogramme zu Datenbanken und zur Informationsbeschaffung (exploratives Lernen),
- Simulations- und Berechnungsprogramme für die Analyse komplexer Zusammenhänge (exploratives Lernen),
- Lernprogramme mit höheren Ansprüchen (elektronischer Tutor oder sogar Pädagoge).

Heute herrschen Lernprogramme aus den ersten vier Kategorien vor. Lernprogramme mit höheren Ansprüchen scheitern zurzeit vor allem (noch?) an der Beschränktheit des Mediums Computer. Der Dialog zwischen Lernendem und Lehrer in einer natürlichen Sprache spielt im Lernprozess eine entscheidende Rolle. Er kann im heutigen Zeitpunkt von keinem Computersystem bewerkstelligt werden.

Lernprogramme gibt es nicht bloss in grosser Zahl, sondern auch in frappant unterschiedlicher Qualität. Das ist eigentlich schade, denn wie *Nievergelt* und *Ventura* [14] gezeigt haben, wäre es schon mit bescheidenen Hilfsmitteln (d.h. cursoradressierbarem zeichenorientiertem Bildschirm) möglich, interaktive Unterrichtsprogramme guter Qualität herzustellen. Obwohl heute praktisch jedes Bildschirmgerät, ob intelligenzloses Terminal oder Arbeitsplatzrechner, diesen Anforderungen zu genügen vermag, machen erstaunlich wenige Unterrichtsprogrammautoren von einer bewussten Gestaltung des Mediums Computerbildschirm Gebrauch. Es scheitern dann nur zu oft didaktisch gute Ideen für Unterrichtsprogramme an einer unbefriedigenden Ausführung. Allerdings mangelt es auch an didaktisch sinnvollen Ideen; nicht alles, was sich auf einem Computer machen lässt, eignet sich auch optimal zu diesem Zweck. Für den Computer gilt genau gleich wie für alle anderen di-

daktisch einsetzbaren, technischen Hilfsmittel, z.B. Dia- oder Filmprojektion, dass er als didaktisches Medium richtig eingeschätzt werden muss. Gewisse Sachverhalte, z.B. dynamische Vorgänge, lassen sich sehr gut oder bloss mit Hilfe eines Computers darstellen, statische Fakten werden jedoch mit Vorteil oft besser durch traditionelle Lehrmedien vermittelt.

Um allfälligen Ängsten vorzubeugen, soll abschliessend darauf hingewiesen werden, dass der Dozent durch computergestützten Unterricht auf keinen Fall ausgeschaltet und überflüssig gemacht werden soll, noch kann. Selbst die besten Lernprogramme können nur ganz spezifische Aufgaben übernehmen. Dort, wo allerdings ein Dozent nur selten zur Verfügung steht (was z.B. für die Weiterbildung öfters zutrifft), kann der Computer einen wesentlichen Beitrag zur wirkungsvollen Förderung eines Selbststudiums leisten.

4. Autorenumgebungen

Autorensysteme sind Softwareentwicklungswerkzeuge, die das Erstellen von Unterrichtsprogrammen erleichtern. Der Möglichkeit, Lernprogramme effizient erzeugen oder zumindest anpassen zu können, kommt eine entscheidende Bedeutung zu. Lehrmittel haben schon immer erst dann weite Verbreitung gefunden, wenn das Medium dem Lehrer erlaubt hat, dem Lehrmittel seinen urpersönlichen Stempel aufzudrücken. Genau das ist ohne Autorensysteme bei Lehrprogrammen meist nicht oder nur mit kaum verantwortbar grossem Aufwand möglich.

Autorensystemen kommt jedoch noch eine weitere, oft zu wenig beachtete Funktion zu. Sie prägen den Mensch-Maschinen-Dialog in entscheidender Weise. Unsere Erfahrung hat gezeigt, dass im praktischen Einsatz von interaktiven Unterrichtsprogrammen der Gestaltung des Benutzerdialoges eine ausschlaggebende Bedeutung zukommt. Ein Lernprogramm mag in seiner Grundidee noch so gut sein, seine Verwendung als Lernmittel wird sinnlos, wenn der Benutzer infolge schlechter Dialoggestaltung all seine Lernzeit vor dem Rechner mit dem Lösen von Dialogproblemen aufbraucht. Zudem ermöglichen es gute Dialogprogramme, das Schwergewicht des Gesprächs zwischen Betreuern (Dozenten, Assistenten) und Studierenden auf den Lehrinhalt statt auf das Lösen betriebstechnischer Probleme zu legen.

Ganz allgemein gilt, dass sich eine optimal erfolgreiche Verwendung von interaktiven Rechnersystemen bloss dann ergibt, wenn ein konzises, leicht verständliches und durch das System konsequent eingehaltenes Benutzermodell vorliegt. Hierzu gehören beispielsweise die Art der Befehlseingabe wie auch die Systemreaktion auf bestimmte Befehle. Nicht die äusserlich eindrücklichen Erscheinungen, wie z.B. das Vorhandensein von Fenstern oder kompliziert aufgebauten Bildschirmen, vollgepfropft mit verwirrender, unnötiger Information, sondern die funktional zweckmässige Verwendung der Benutzerschnittstellenelemente, z.B. Menüs, ist entscheidend. Der Befehlssatz muss möglichst klein sein, und bewährt hat sich eine ständig präsente oder zumindest jederzeit aufrufbare Gedächtnisunterstützung in bezug auf den momentan gültigen Befehlssatz (einblendbare Rollbalken oder Pop-up-Menüs) und die explizite oder zumindest implizite Anzeige des momentanen Systemzustandes.

Nievergelt [15] formuliert die Anforderungen an ein interaktives System wie folgt: Der Benutzer muss zu jedem Zeitpunkt Antwort auf die folgenden Fragen bekommen können:

- Wo bin ich?
- Was kann ich hier tun?
- Wie kam ich hierher?
- Wo kann ich von hier aus hingehen und wie stelle ich das an?

Leider gestatten viele der zurzeit beliebtesten Arbeitsplatzsysteme gerade die Beantwortung der Mehrheit dieser Fragen nicht, ohne dass z.B. die gerade laufende Tätigkeit unterbrochen werden muss.

Im Rahmen des Pilotprojektes CELTIA (*C*omputerunterstütztes *E*xploratives *L*ehren und *T*rainieren mit *I*nteraktiv *A*nimierter Simulation), mit dem an der ETH Zürich Erfahrungen über die mathematische Modellierung und Simulation dynamischer Systeme gesammelt werden, wurde als Teil einer Autorenumgebung ein Softwareprodukt entwickelt, das viele der angesprochenen Punkte berücksichtigt. Die Dialogmaschine [16] entspricht einer abstrakten Maschine, die ein einfaches, klar definiertes und leicht verständliches Benutzermodell anbietet. Mensch-Rechner-Dialoge werden ausschliesslich unter Kontrolle der Dialogmaschine und unter gleichzeitiger Verwaltung der Dialogschnittstellenelemente (Menüs, Arbeitsfenster, Dialogkasten, Anzeigefenster, Zeigefelder, Tasten, Rollbalken, Druckknöpfe und Editierfelder für alle elementaren Datentypen usw.) durchgeführt. Dadurch wird nicht nur die äusserlich sichtbare Erscheinungsform der Programme, sondern auch deren Verhalten vereinheitlicht.

Ihre Funktionalität erzielt die Dialogmaschine dadurch, dass sie sich zwischen das eigentliche Lernprogramm und den Benutzer schiebt. Durch die Auslösung eines sog. Benutzerereignisses, z.B. einer Menüauswahl, erfolgt eine Befehlseingabe, die zunächst von der Dialogmaschine abgefangen und nach Möglichkeit automatisch beantwortet wird. Diese Standardreaktionen der Dialogmaschine sind Teil des dem Benutzer bekannten Benutzermodells, z.B. Bewegen oder Schliessen eines Fensters oder Erfassen und Verschieben eines grafischen Objektes infolge bestimmter Tätigkeiten durch den Benutzer am System. Lediglich über genau festgelegte, dem Programmierer bekannte Schnittstellen können diese Benützerereignisse an die lernprogrammspezifischen Programmteile weitergeleitet werden, wo sie eine Aktion des Lernprogrammes, z.B. eine numerische Integration, auszulösen vermögen. Die Dialogmaschine hat also die Funktion eines Filters, das sich in einer ständigen Empfangsbereitschaft für Benutzerereignisse befindet, die nach Möglichkeit durch deren sofortige Behandlung eine vordefinierte Reaktion auslösen. Erst anschliessend können die Benutzerereignisse an das eigentliche Lernprogramm gelangen, wo sie eine nicht standardisierbare Programmaktion auszulösen vermögen.

Die folgenden Benutzerereignisse werden durch die Dialogmaschine definiert: Drücken der Tastatur oder eines Mausknopfes. Da die Maus ein Zeigegerät ist, ist es naheliegend, das Drücken des Mausknopfes kontextabhängig weiter zu definieren. Hierbei muss unterschieden werden zwischen Benutzerereignissen, die eine lediglich vordefinierte Standardreaktion der Dialogmaschine zur Folge haben können, z.B. das Verschieben eines Fensters innerhalb des Bildschirmes oder das Betätigen eines Rollbalkens, und denjenigen Benutzerereignissen, die ebenfalls eine lernprogrammspezifische, nicht standardisierte Aktion auszulösen imstande sind. Sie dienen der eigentlichen Befehlseingabe und be-

wirken oft eine Transition von einem Programmodus in den andern. Sie betreffen das Einblenden und Auswählen eines Menübefehls, das Schliessen oder die Grössenveränderung eines Fensters, die Aktivierung einer Drucktaste in einem Arbeitsfenster oder in einem eigenen Dialogkasten und das Drücken einer Maustaste innerhalb eines Fensters. Alles übrige Betätigen der Maustasten bleibt ohne irgendwelche Systemreaktion.

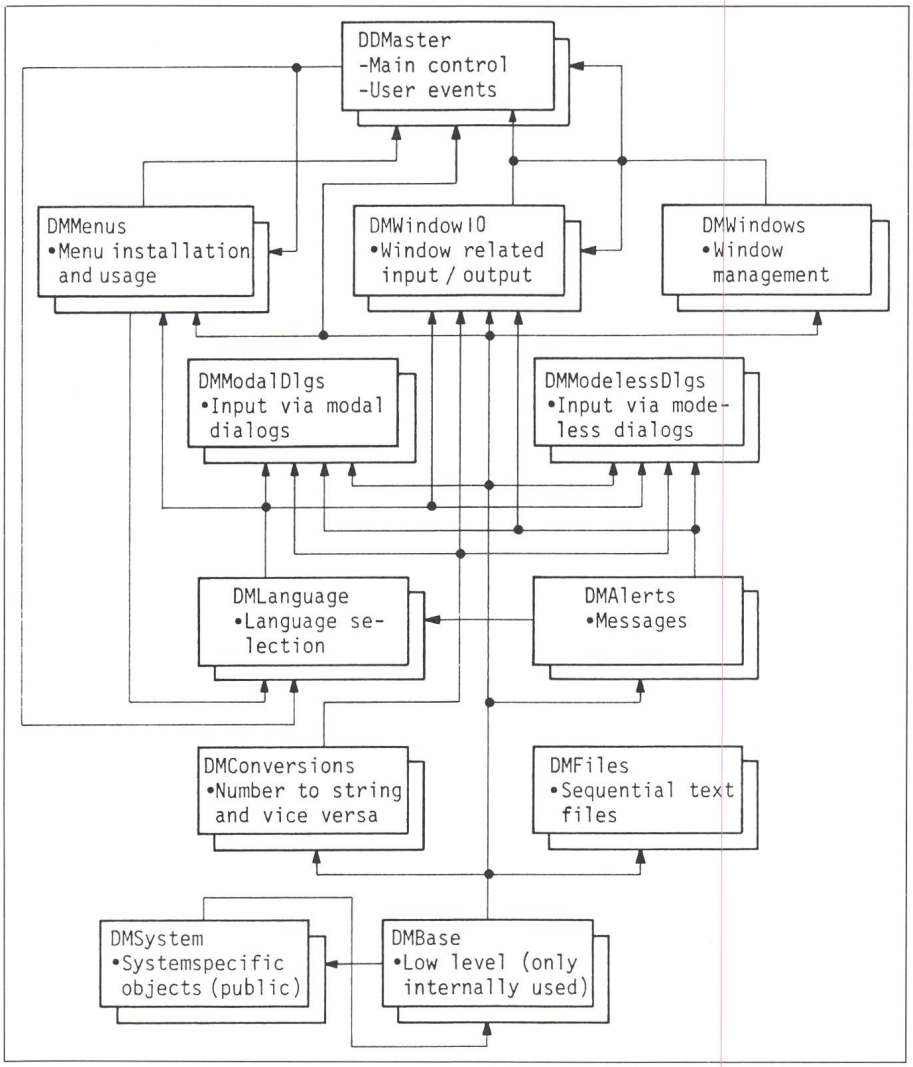
Als weitere Konsequenz ergibt sich aus diesem Ansatz, dass ein mit der Dialogmaschine erstelltes Programm gar keine eigentliche Flusskontrolle mehr aufweist. Ein typisches Dialogmaschinenprogramm besteht lediglich aus zwei Anweisungen: Der Installation des Befehlssatzes in Form der Deklaration von einem Satz von Prozeduren, wobei für jeden Befehl mindestens eine Prozedur mitdeklariert werden muss, und der Übergabe der Programmkontrolle an die Dialogmaschine.

Die Dialogmaschine verwirklicht dadurch einen endlichen Automaten[2], wobei jeder Automatenzustand eindeutig einem bestimmten Programmodus entspricht. Befehle dienen dazu, Transitionen von einem zum anderen Zustand zu ermöglichen. Der Programmierer sollte lediglich darauf achten, dass Transitionen von einem möglichst augenfälligen Ereignis, das auf dem Bildschirm grafisch dargestellt wird, begleitet werden, z.B. dem Erscheinen oder Schliessen eines grossen Fensters oder einer Aufmerksamkeit erheischenden grafischen Ausgabe. Damit erfolgt eine natürlich verständliche Selbstdokumentation des Systemzustandes, die es erlaubt, die Frage «Wo bin ich?» zu beantworten.

Die Frage «Was kann ich hier tun?» muss durch eine Befehlspalette beantwortbar sein, die ständig in einem eigens dafür reservierten Bildschirmteil z.B. in Form von einblendbaren Menüs und den dazugehörigen auswählbaren Befehlen auch dann präsent ist, wenn keine Befehlsauslösung gemacht wird. Eine allfällige Befehlshierarchie kann durch entsprechende Untermenüs oder eigens aufrufbare Dialogkasten verwirklicht werden.

Jeder Zustand sollte jedoch von jedem anderen Systemzustand aus erreicht werden können, um die Frage

[2] Ein Automat, der nur eine endliche Anzahl Zustände kennt.



Figur 2. Modulstruktur der Dialogmaschine [16]

«Woher kam ich?» belanglos zu machen. Falls das einmal ausnahmsweise nicht bewerkstelligt werden kann, so muss der Programmierer zumindest dafür sorgen, dass der vorherige und gegenwärtige Systemzustand ohne grosse Mühe von dem zurzeit vorhandenen Befehlssatz und den auf dem Bildschirm sichtbaren Objekten abgeleitet werden kann.

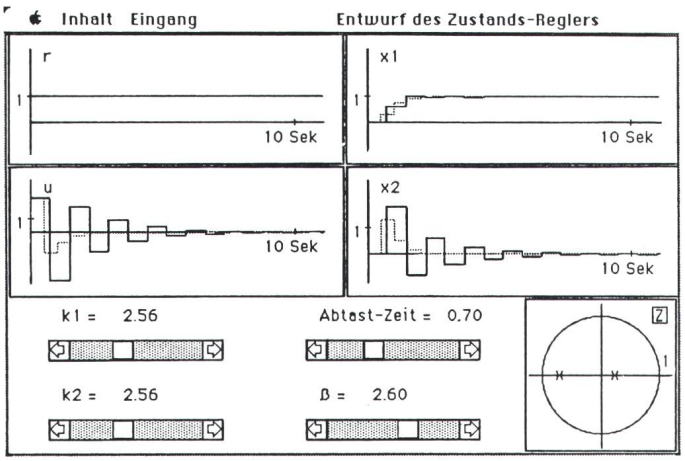
Transitionen haben möglichst schnell abzulaufen, oder dann muss zumindest eine Anzeige «Alles ist in Ordnung, jedoch bitte warten» an den Benutzer erfolgen. Nach Ablauf der Transition erfolgt die Rückgabe der Programmkontrolle an die Dialogmaschine automatisch.

Die Dialogmaschine ist vorerst für den Apple Macintosh, die an der ETH Zürich zurzeit weitverbreitetste Unterrichtsmaschine, implementiert worden. Sie besteht aus einem Satz von Modula-2-Modulen. Die Dialogmaschine ist zur Programmierung mehrerer Lernprogramme an verschiedenen Instituten der ETH Zürich erfolgreich verwendet worden. Die Modulstruktur, die gemäss der Benutzerschnittstellenelemente, über die sich der Dialog abwickelt, gestaltet wurde, ist in Figur 2 dargestellt [16].

Der erwartete positive synergistische Effekt, dass das Programmverhalten vereinheitlicht und trotzdem der Programmieraufwand massiv verkleinert werden kann, hat sich hierbei bestätigt. Dank der Dialogmaschine konnten selbst eher ungeübte Programmierer erfolgreich gute Unterrichtsprogramme herstellen. Im nächsten Abschnitt werden zwei Lernprogramme, die mit Hilfe der Dialogmaschine geschrieben worden sind, kurz vorgestellt. Der Aufwand zu ihrer Entwicklung ohne die Dialogmaschine wäre, da viel zu gross, wohl kaum zu rechtfertigen gewesen.

Die Dialogmaschine erleichtert die Entwicklung von Lernprogrammen,

die der Kategorie der Simulations- und Berechnungsprogramme zur Analyse komplexer Zusammenhänge angehören, wesentlich. Eigentliche Programmgeneratoren sind in diesem Bereich im Gegensatz zum programmierten Unterricht noch kaum vorhanden, jedoch sind Generatoren für sog. Rahmenprogramme erhältlich [17; 18]. Weiterentwicklungen von Autorenumgebungen im Hinblick auf Lernprogramme mit höheren Ansprüchen könnten schliesslich zum elektronischen Buch von *Kohlas* [19] und *Pasquier* [20] oder zum Projekt KOFIS «Prolog-Werkzeuge für eine integrierte Dokumenten- und Wissensverwaltung» [21] führen.



Figur 3
Ein typischer Arbeitsbildschirm, wie er sich dem Studenten beim Betreiben des Unterrichtsprogramms «Zustandsregler» [24] präsentiert

5. Beispiele von Lernprogrammen

5.1 Das Unterrichtsprogramm «Zustandsregler»

In der Vorlesung *Automatisierungstechnik* erhalten die Studierenden Gelegenheit, Zustandsregler zu entwerfen und zu simulieren. Die Theorie ist in [22] dargestellt. Die Entwürfe können mit MATLAB [23] ausgeführt werden und die Simulationen mit dem Programm «Zustandsregler» [24]. Die Figur 3 zeigt einen typischen Arbeitsbildschirm für die Studierenden. Die Erstellung des Programms in Modula-2 unter Verwendung der Dialogmaschine beanspruchte 3 Mannwochen.

5.2 Das Unterrichtsprogramm «Stabilität»

In der Vorlesung «Systemanalyse», bei der die Analyse und Simulation ökologischer Systeme im Vordergrund steht, haben sich die Studenten der Naturwissenschaften und der Landwirtschaft mit den Begriffen der Stabilität von Gleichgewichtslagen auseinanderzusetzen. Hierzu erhalten sie Vorlesungs- und Übungsunterlagen, die es ihnen gestatten, anhand einer Analyse des zeitlichen Verhaltens Korrespondenzen zwischen den Stabilitätseigenschaften dreier in der Natur beobachteter, dreier mathematischer Modellsysteme und der drei durch das Lernprogramm zur Verfügung gestellten Simulationsmodelle aufzudecken. Die Lösung dieser Aufgabe ist nur möglich bei richtiger Erkennung des Stabilitätsverhaltens der einzelnen Systeme und dem klaren Verständnis der einzelnen Stabilitätsbegriffe. Um die Arbeit übersichtlich und einprägsam zu gestalten, ist aufbauend auf der Dialogmaschine ein grafischer Editor für das Verändern eines dreidimensionalen Koordinatensystems geschrieben worden, der dem Studenten gestattet, die verschiedensten Sichten auf das Simulationsgeschehen wählen zu können (Fig. 4). Er kann beispielsweise frei zwischen der Darstellung des zeitlichen Verlaufes oder der Zustandsraumdarstellung wählen und dadurch auf einfache Weise auch Phasenportraits erzeugen [25]. Zudem können die Modellsysteme verschiedensten Störungen unterworfen und Modellparameter sowie Anfangswerte abgeändert werden. Die bisherigen Rückmeldungen der Studenten waren äusserst positiv, und der Lernerfolg ist sehr zufriedenstellend ausgefallen. Zusammen mit der Entwicklung des allgemein einsetzbaren grafischen Koordinatensystemeditors und einer Laufzeitbibliothek für Simulationen (verschiedene Verfahren der numerischen Integration) dauerte die Entwicklung des Programms «Stabilität» 7 Mannwochen.

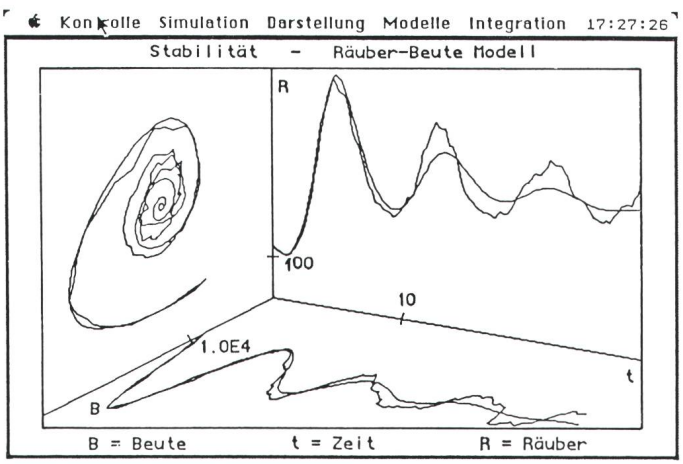
5.3 Methoden der grafischen und geometrischen Datenverarbeitung

Für diesen zentralen Bereich des Computer Aided Design wurde am Institut für Informatik der ETHZ das Unterrichtssystem POLY (Meier/Loacker 1986) mit der folgenden Zielsetzung geschaffen:

- POLY ist ein Lehr- und Lernsystem für Computergrafik und Computergeometrie.
- POLY ist ein geometrisches Modelliersystem für räumliche Objekte.
- POLY ist ein Test- und Experimentierfeld für dreidimensionale Problemstellungen.

6. Schlussbemerkung

Lernprogramme können heute in den Kategorien Test- und Prüfungsprogramme, Drill- und Trainingsprogramme sowie Simulationsprogramme besonders erfolgreich eingesetzt werden. Insbesondere Simulationsprogramme sind interessant, da sie ein ex-



Figur 4
Ein typischer Arbeitsbildschirm, wie er sich dem Studenten beim Betreiben des Unterrichtsprogramms «Stabilität«» [25] präsentiert

ploratives Lernen ermöglichen können, das bis anhin ohne Computer kaum denkbar war. Ein besseres Verständnis der Studierenden für komplexe Zusammenhänge, insbesondere bei dynamischen Systemen, lässt sich so ohne weiteres erzielen. Zumindest gemäss unserer Erfahrungen haben sich Simulationsprogramme nicht bloss als entscheidend für den computerabhängigen Unterricht erwiesen, sondern haben sich auch für den computerunterstützten Unterricht als neues didaktisches Medium zwecks Vermittlung beliebigen Lehrstoffes bewährt.

Die Herstellung von spezifischen Unterrichtsprogrammen, die genau in eine bestimmte Unterrichtsveranstaltung passen, ist besonders attraktiv, jedoch mit grossem Aufwand verbunden. Er kann auf ein vertretbares Mass heruntergedrückt werden, indem Autorensysteme, z.B. in der Art der Dialogmaschine, zum Einsatz kommen. Dies bringt zudem den wichtigen Vorteil zustande, dass das Verhalten der Unterrichtsprogramme vereinheitlicht und die konsequente Verwendung eines leicht verständlichen Benutzermodells dadurch automatisch garantiert werden kann.

Schlussendlich sind nicht die Hilfsmittel entscheidend, sondern eine geeignete didaktisch einsetzbare Idee und deren klare, zielgerichtete Verwirklichung in Form eines Unterrichtsprogrammes. Hierzu ist ein gehöriges Mass an einschlägiger Erfahrung vonnöten, das an den meisten schweizerischen Hochschulen noch gesammelt werden muss. Erst die Kombination von didaktischem Einfallsreichtum, der auf guten Fachkenntnissen beruht, mit entsprechender Erfahrung am neuartigen Medium Arbeitsplatzrechner wird neue Unterrichtssoftware wie auch neuartige Unterrichtsformen entstehen lassen. Beispielsweise ist denkbar, dass der Studierende bereits in naher Zukunft regelmässig während eines grossen Teils seiner Studienzeit selbständig und frei mit technischen und naturwissenschaftlichen Systemen experimentieren wird. Bei einem solchen entdeckenden Lernen haben Unterrichtsprogramme in Form von Simulationsprogrammen eine wichtige Rolle zu spielen. Beispielsweise erbringen sie den Vorteil, dass der Student nicht gleich zu fürchten braucht, dass die am realen System früher oder später unausweichliche Katastrophe beim Erforschen der verschiedensten, auch naiven Szenarien und Einflüsse, Wirklichkeit werden könnte. So könnte bei vielen Hochschulabsolventen ein reifes und vertieftes Verständnis komplexen Sach- und Systemverhaltens gefördert werden, was nicht nur dem Studierenden zugute käme, sondern auch die Forderungen der Gesellschaft an den technisch-naturwissenschaftlichen Hochschulunterricht besser berücksichtigen dürfte.

Literatur

[1] *J. Nievergelt, A. Ventura* and *H. Hinterberger:* Interactive computer programs for education - philosophy, techniques and examples. Reading/Massachusetts a.o., Addison-Wesley, 1986.

[2] *T.F.M. Stewart:* Professional workstations. - State of the art report - Maidenhead, Pergamon Infotech, 1983.

[3] *H.-J. Mey:* Der Eintritt in das Informationszeitalter. In: H. Keller: Denken über die Zukunft. Zürich, Ringier-Verlag, 1986; S. 82...99.

[4] *C. A. Zehnder:* Computerarbeitsplätze für jedermann - im Verbund. Kommunikation (1985)2, April, S. 37...42.

[5] *F.E. Cellier:* Simulation software: Today and tomorrow. SGA-Zeitschrift 4(1984)1, p. 7...22.

[6] *M. Rimvall, M. Mansour* and *W. Schaufelberger:* Computer-aided design of control systems, an integrated approach. Proceedings of the third IFAC Symposium on Computer-Aided Control System Design. Oxford a.o., Pergamon Press, 1985.

[7] *W. Schaufelberger, P. Sprecher* und *P. Wegmann:* Echtzeit-Programmierung bei Automatisierungssystemen. Stuttgart, B.G. Teubner, 1985.

[8] *M. Polke:* Informationshaushalt technischer Prozesse. ATP Automatisierungstechnische Praxis 27(1985)4, S. 161...171.

[9] *H.-J. Appelrath:* Von Datenbanken zu Expertensystemen. - Informatik-Fachberichte 102 - Berlin u.a., Springer-Verlag, 1985.

[10] *H. Nettesheim:* Programmentwurf und Programmdokumentation. Methoden und Techniken bei der Prozessdatenverarbeitung. Düsseldorf, VDI-Verlag, 1982.

[11] *C.A. Zehnder:* Informatik-Projektentwicklung. Eine Einführung für Informatikstudenten und Praktiker. Zürich, Verlag der Fachvereine, 1986.

[12] *H. Fischer:* Allgemeine Didaktik für höhere Schulen. Zürich, Verlag der Fachvereine, 1983.

[13] *E. Forte:* Les langages-auteur d'enseignement assisté par ordinateur. Lausanne, Ecole Polytechnique Fédérale, Laboratoire de Microinformatique, 1985.

[14] *J. Nievergelt* und *A. Ventura:* Die Gestaltung interaktiver Programme; mit Anwendungsbeispielen für den Unterricht. Stuttgart, B.G. Teubner, 1983.

[15] *J. Nievergelt:* Errors in dialog design and how to avoid them. In: Document preparation systems. Edited by J. Nievergelt a.o. Amsterdam a.o., North-Holland, 1982, p. 265...274.

[16] *A. Fischlin:* Simplifying the usage and the programming of modern working stations with Modula-2: The Dialog Machine. Zürich, Eidgenössische Technische Hochschule, Institut für Automatik und Industrielle Elektronik, 1987.

[17] *A. Ventura:* Einsatz und Programmierung des Computers als Werkzeug für den Unterricht. Dissertation Nr. 7752 der Eidgenössischen Technischen Hochschule Zürich, 1985.

[18] *K. Vancso:* Users models of interactive learning programs and their realization aspects. Zürich, Eidgenössische Technische Hochschule, Institut für Automatik und Industrielle Elektronik, 1986.

[19] *J. Kohlas:* Das integrierte Buch (eine Projektidee). Fribourg, Universität, Institut für Automation und Operations Research, 1984.

[20] *J.A. Pasquier:* Les avantages du livre électronique (rapport sur un projet de recherche). Fribourg, Universität, Institut für Automation und Operation Research, 1985.

[21] *H.J. Appelrath* u.a.: Das Projekt KOFIS: Prolog Werkzeuge für eine integrierte Dokumenten- und Wissensvermittlung. Zürich, Eidgenössische Technische Hochschule, Institut für Informatik, 1986.

[22] *W. Schaufelberger:* Ein allgemeines Vorgehen für den Entwurf von Zustandsreglern. Scientia Electrica 31(1985)1, S. 1...22.

[23] *C. Moler:* Matlab, user's guide, Albuquerque/USA, University of New Mexico, Department of Computer Science, 1980.

[24] *A. Itten:* Unterrichtsprogramm: Diskrete Zustandsregler mit Beobachter. Zürich, Eidgenössische Technische Hochschule, Institut für Automatik und Industrielle Elektronik, 1986.

[25] *A. Fischlin* und *M. Ulrich:* Stability: An educational program for students of systems ecology. Zürich, Eidgenössische Technische Hochschule, Institut für Automatik und Industrielle Elektronik, 1987.

[26] *A. Meier* und *H. Loacker:* Poly - Ein Unterrichtssystem zur Beschreibung, Darstellung und Manipulation ebenbegrenzter Objekte. Zürich, Eidgenössische Technische Hochschule, Institut für Informatik, 1986.

[27] *A. Fischlin* a.o.: Simulation and computer aided control system design in engineering education. IFAC/IMACS International Symposium on Simulation and Control Systems, Vienna, 1986.

[28] *A. Meier:* Methoden der grafischen und geometrischen Datenverarbeitung. Stuttgart, G.B. Teubner, 1986.

[29] *C.A. Zehnder:* Informationssysteme und Datenbanken. 3. Auflage. Zürich, Verlag der Fachvereine, 1985.