HEINRICH - HERTZ - INSTITUT — BERLIN-CHARLOTTENBURG

Technischer Bericht Nr. 200

PROGRAMM - VORFAHRT

ERSTER VERSUCH DER IMPLEMENTIERUNG EINES LERNERADAPTIVEN LEHRSYSTEMS (LAL) IM ZWEIWEG - KABELFERNSEHSYSTEM DES HHI

von

Dr.-Ing. Günther Creutz
Dipl.-Päd. Lothar Mühlbach

Berlin

1978

EINSTEINUFER 37 1000 BERLIN 10

HEINRICH-HERTZ-INSTITUT FÜR NACHRICHTENTECHNIK BERLIN
Gesellschaft mit beschränkter Haftung

Heinrich-Hertz-Institut, Einsteinufer 37, 1000 Berlin 10

HHI

Technischer Bericht Nr. 200

PROGRAMM VORFAHRT

Erster Versuch der Implementierung eines lerneradaptiven Lehrsystems (LAL) im Zweiweg-Kabelfernsehsystem des HHI

Zusammenfassung

Der vorliegende Bericht bezieht sich auf Probleme der Implementierung adaptiver Dialogsysteme mit wechselnder Initiative in einem Zweiweg-Kabelfernsehsystem (ZKTV).

Durch die Implementierung des Programms VORFAHRT inform eines lerneradaptiven Lehrsystems (LAL) sollte u.a. die Frage beantwortet werden, welche Funktionen nach Festlegung eines bestimmten Qualitätsstandards der Kommunikation im HHI-Laborsystem z.Zt. noch nicht hinreichend effektiv und/oder benutzerfreundlich vom Rechner erfüllt werden können. Die Interaktionen des Teilnehmers mit dem ZKTV sollte deshalb einem menschlichen Dialog möglichst angenähert werden. Um das zu gewährleisten, konnte der Lerner beispielsweise mit einem menschlichen Berater (Tutor) in der Zentrale kommunizieren, wenn seine Fragen oder Wünsche über den vorgegebenen Rahmen des Lehrprogramms hinausgingen.

Bearbeiter:

Dr. G. Creutz
(Dr. Günther Creutz)

(Dipl.-Päd. Lothar Mühlbach)

WISS.TECHN.GESCHÄFTSFÜHRER
(Dr. Horst Ohnsorge)

ABTEILUNGSLEITER
(Dr. Karl-Hinrich Vöge)

Berlin-Charlottenburg, den 6.2.1978

Inhalt:

Anhang

1 Mediendidaktische Konzeption eines LALs

Mit Begriffen wie "Adaptiver Unterricht"(z.B. Schwarzer, Steinhagen 1975), "Lerner-adaptives Lehrsystem" (Issing et al., 1977), oder "Lernerorientiertes Informationssystem"(Haefner 1977) kann man erst dann sinnvoll operieren, wenn man weiß, welche Systemphilosophie sich hinter ihnen im einzelnen verbirgt.
Am Anfang dieser Dokumentation über das Programm VORFAHRT soll deshalb kurz auf die mediendidaktische Konzeption eines lerneradaptiven Lehrsystems (LAL) eingegangen werden, wie sie von der Projektgruppe "Individuelles interaktives Lernen" (Issing et al., 1977) unter der Zielsetzung einer späteren Implementierung in einem Zweiweg-Kabelfernsehen (ZKTV) entwickelt wurde.

1.1 Computerunterstützter Unterricht und Bildungsfernsehen

Nachdem die begrenzten Möglichkeiten von tutoriellen Programmen im Computerunterstützten Unterricht (CUU) von der überwiegenden Mehrzahl der in diesem Bereich arbeitenden Wissenschaftlern erkannt wurden (vgl. Eyferth et al. 1974, S. 113ff), fehlt zunächst ein theoretisches Modell, das die Frage nach dem sinnvollen Einsatz von Rechnern in solchen Lehr-Lernsituationen, die vornehmlich der Aneignung neuen Wissens dienen, in einem für die Praxis hinreichendem Maße beantworten kann.

Diese Schwierigkeit kann man scheinbar dadurch umgehen, daß man das Postulat der "Lehrobjektivierung" (vgl. z.B. Frank/Meder 1971) nicht mehr generell erhebt, sondern vorrangig die Formen des CUU weiter ausbaut, die sich auch ohne umfassende theoretische Grundlegung als praktizierbar herausgestellt haben (z.B. Übungsprogramme, Simulation und Spiel, interaktives Programmieren).

Daß für das Aneignen neuen Wissens und das Anbieten gezielter Lernhilfen erst ansatzweise neue Modelle existieren, die sich auf einer Rechenanlage implementieren lassen, stellt sich für den "normalen Schulunterricht" solange nicht als Mangel dar, wie die Simulation eines "Privatlehrers" aufgrund des Vorhandenseins genügend menschlicher Lehrer unnötig erscheint.

Auch die herkömmlichen Programme des Bildungsfernsehens müssen sich nicht auf ein explizites Modell von komplexen Lehr-Lernsituationen stützen, da ein wesentliches Element solcher Situationen - die Interaktion - bei ihnen nahezu ausgeschlossen ist.
In der "Praxis des Unterrichts" geht man häufig davon aus, daß ein menschlicher Lehrer die gewünschten Lernprozesse beim Lernenden dann in ausreichendem Maße initiieren kann, wenn er über gewisse "Erfahrung" sowie über die Möglichkeit des Einsatzes verschiedener "einfacher" Medien verfügt.
Diese Auffassung könnte sich u.a. darauf berufen, daß Lehren schon seit einigen tausend Jahren auch ohne ein exaktes Modell des Unterrichts und ohne komplexe und komplizierte Mediensysteme (vgl. Döring 1973) anscheinend funktioniert.

Betrachtet man die eben angerissenen Probleme jedoch näher, so stellt sich heraus, daß das Erstellen einer "Theorie des Unterrichts" nicht nur eine Frage einer, von der Unterrichtspraxis weit entfernten, didaktischen Grundlagenforschung ist.

In Hinsicht auf eine Implementierung von Lehrsystemen in einem ZKTV stellt sich die Frage nach einer möglichst exakten Modellbildung unter einem ganz bestimmten Gesichtspunkt.
Hier kann nicht davon ausgegangen werden, daß es für den zu erteilenden Unterricht von vornherein einen oder mehrere Lehrer gibt und sich dann die Frage stellt, welche der von ihnen ausgeübten Funktionen von einem Rechner ausgeführt werden können.
In gewissem Sinne muß man im Falle des ZKTV genau vom entgegengesetzten Ausgangspunkt her fragen:

> Welche Funktionen in den gewünschten Diensten (hier hauptsächlich im Bildungsbereich) können nach Festlegung eines bestimmten Qualitätsstandards der Kommunikation von einem Rechnersystem nicht hinreichend effektiv und/oder benutzerfreundlich erfüllt werden, sodaß auf einen menschlichen Lehrer oder Berater nicht verzichtet werden kann?

Eine Modellbildung, die für die Beantwortung dieser Frage hinreichend exakt ist, muß keine "umfassende" Theorie "des Unterrichts" oder "des Lehrens" sein.

Eine solche Modellbildung muß aber die in einem ZKTV ablaufenden Interaktionsprozesse so exakt abbilden, daß ein kontrolliertes Verändern von Variablenwerten zum Zwecke der Evaluierung vorhandener Interaktionsstrukturen und damit die Beantwortung der Frage nach effektiven und benutzerfreundlichen Diensten möglich wird.

1.2 Lerner-Tutor-Dialoge

Auf welche theoretische Basis kann sich aber eine Modellbildung für Lehr-Dienste stützen, wenn mit der Kritik am Behaviorismus auch dem herkömmlichen tutoriellen CUU - als einem "Kind der behavioristischen Lernforschung" (Eyferth et al. 1974) - weitgehend eben diese Basis entzogen wurde?

Die Projektgruppe "individuelles interaktives Lernen" (Issing et al. 1977) sieht für die Konzipierung eines LALs im ZKTV eine akzeptable Grundlage in einem theoretischen Ansatz, der weitgehend die Kritik am behavioristischen Lernmodell vorangetrieben hat, der "zwischen Reiz und Reaktion ein bißchen Weisheit" (Miller et al. 1974) einschieben will: der "kognitiven Psychologie".

Beim menschlichen Lernen geht es, im Unterschied zum tierischen Lernen, um die Übermittlung und Verarbeitung bedeutungshaltigen Wissens (vgl. z.B. Leontjew 1973, S. 218ff). Nur solche Theorien können Hinweise für die teilweise Objektivierung der Funktionen eines menschlichen Tutors liefern, die etwas darüber aussagen, wie solches Wissen im Gedächtnis repräsentiert wird und wie die kognitiven Strukturen aktualisiert, modifiziert und zum Erwerb neuen Wissens benutzt werden.

Hierfür bieten die Ansätze der kognitiven Psychologie einige brauchbare Hinweise; ein Unterschied zur behavioristischen Lerntheorie besteht darin, daß sie den menschlichen Organismus als einen von sich aus aktiven Ver- und Bearbeiter von Wissen auffaßt. - "Lernen wird nicht mehr nur als Reaktion auf Lehren verstanden" (Dohmen 1976, - diese Bemerkung bezieht sich bei Dohmen auf eine Individualisierung des Lernens durch Medieneinsatz).

Hieraus ergeben sich auch die - von herkömmlichen tutoriellen Programmen unterscheidbaren - Implikationen für adaptive Lehrsysteme auf der Grundlage kognitiver Theorien:
Der Lehrer (oder ein technisches System, das Lehrfunktio objektiviert) hat nicht primär die Aufgabe der Verstärku (im Sinne der behavioristischen Lerntheorie) sondern die der Unterstützung bei der Integration neuen Wissens in die vorhandene kognitive Struktur des Lernenden.
Dadurch bekommt auch der Begriff der Interaktion zwischen Lehr- und Lernsystem eine andere Bedeutung als im traditionellen tutoriellen CUU.
Ging es dort primär darum, dem Lernenden ein feedback über die Richtigkeit seiner Antwort zu übermitteln, so dient die Interaktion bei einem auf kognitions-psychologischer Grundlage konzipierten Lehrsystem auch dazu, Hypothesen über die kognitive Struktur des Lerners zu bilden und zu evaluieren, möglichst optimal an diese Struktur anzuknüpfen, das Lernmaterial entsprechend zu organisieren, den Auflösungsgrad von Informationseinheiten festzulegen, Wissenslücken zu füllen, Veranschaulichung auf die Codierungsschemata zu beziehen und den Lerner zur Reflexion über seine Lernstrategien anzuregen.

Damit ein adaptives Lehrsystem alle diese vielfältigen Aufgaben erfüllen kann, sollte die Interaktion mit dem Lerner möglichst "offen", d.h. einem menschlichen Dialog möglichst angenähert sein.

Ein solches System realisiert sich beispielsweise im Gespräch eines Nachhilfelehrers mit seinem Schüler; das Konzept solcher Lerner-Tutor-Dialoge mit wechselnder Initiative kann aber auch als Modell für ein adaptives Lehrsystem aufgefaßt werden, in dem sowohl ein Mensch als auch ein Rechner bestimmte Funktionen übernehmen.

Wenn man feststellen kann, daß von der kognitiven Theorie schon wesentliche Impulse für die Weiterentwicklung von Lerntheorien kamen, so darf doch nicht vergessen werden, daß eine Lehrtheorie bzw. -strategie (und diese braucht man für die Objektivierung von Lehrfunktionen) nicht einfach aus einer Lerntheorie oder aus ihren Ansätzen abgeleitet werden kann.

In unserem Zusammenhang interessant sind beispielsweise die Untersuchungen von Collins et al. (1975), die mit der Methode der "Dialog Analysis" der Frage nachgingen: "How does the tutor adapt his teaching to the individual student?" (S. 49). Im Sinne ihrer Zielsetzung - der Modellierung von Tutorfunktionen im Programm SCHOLAR - untersuchten sie dabei hauptsächlich spezielle Probleme von Tutorstrategien wie "Topic Selection", "The Interweaving of Questioning and Presentation" und "The Tutors' Response to Errors" (S. 56ff).
Diese und ähnliche Forschungen sollten vom ZKTV-Laborsystem aufmerksam verfolgt werden.

2 Implementierung eines LALs im ZKTV

Für ein Zweiweg-KTV-System stellt sich u.a. die Frage, ob es möglich und/oder wünschenswert ist, daß alle Funktionen, die ein menschlicher Tutor auszuführen imstande ist, von einem Rechner übernommen werden sollen oder ob der Rückgriff auf einen Menschen in Hinsicht auf Aspekte wie Benutzerfreundlichkeit und/oder Effektivität eines solchen Systems von vornherein als Möglichkeit vorgesehen werden soll.

Zumindest für die ersten Implementierungsversuche wird man (auch zum Zwecke der erwähnten Modellbildung) davon ausgehen müssen, daß ein Mensch bestimmte Funktionen in den interaktiven Bildungsdiensten des ZKTV übernimmt.

> Obwohl in diesem Zusammenhang von Bildungs- bzw. Lernprogrammen die Rede ist, gehen wir davon aus, daß die Kenntnis der in diesen Programmen eruierten Interaktionsstrukturen auch zur Entwicklung anderer ZKTV-Dienste notwendig ist.

Wenn man den oben genannten Konkretisierungen des Begriffs "lerneradaptives Lehrsystem" folgt, dann stellt sich die "praktische" Frage, wie nun das Zusammenspiel zwischen menschlichem Tutor bzw. Berater, Lernprogramm und Benutzer konkret organisiert werden kann.
Dies in einem ersten praktischen Versuch zu erproben, war Aufgabe der zu beschreibenden Implementierung des Programms VORFAHRT.

2.1 Funktionale Komponenten eines LALs

Abstrahiert man sowohl vom Lerninhalt als auch von den materiellen Trägern der Lehr-Funktionen, so kann man feststellen, daß jedes LAL die Eingaben des Lerners aufnimmt, intern verarbeitet und aufgrund interner Algorithmen bestimmte Ausgaben an den Lerner zurücksendet.
Zur Implementierung eines LALs reichen diese allgemeinen Aussagen natürlich noch nicht aus.
Notwendig ist vielmehr die möglichst detaillierte Kenntnis der internen Verarbeitungsprozesse eines solchen Systems, das, soll es im geschilderten Sinne optimal adaptiv sein, nur als ein lernendes System vorstellbar ist.
Die zu betrachtende Lernsituation enthält demnach als entscheidendes Merkmal die Interaktion zweier lernender Systeme - des Lerners und des LALs.
Jedes System lernt dabei durch Rückkopplung der Wirkung seiner Ausgaben auf das andere System (Regelkreismodell).

In einer ersten Konkretisierung des LALs kann man zunächst einige funktionale Komponenten hinsichtlich der internen Verarbeitungsprozesse und der möglichen Ausgaben eruieren (vgl. Abb. 2.1), die für ein solches System konstitutiv sind (Beschreibung der Komponenten: Issing et al. 1977).

Die Möglichkeit, bestimmte Funktionen des LALs an einen Rechner zu delegieren, ist u.a. abhängig von der weiteren Konkretisierung dieser funktionalen Komponenten.

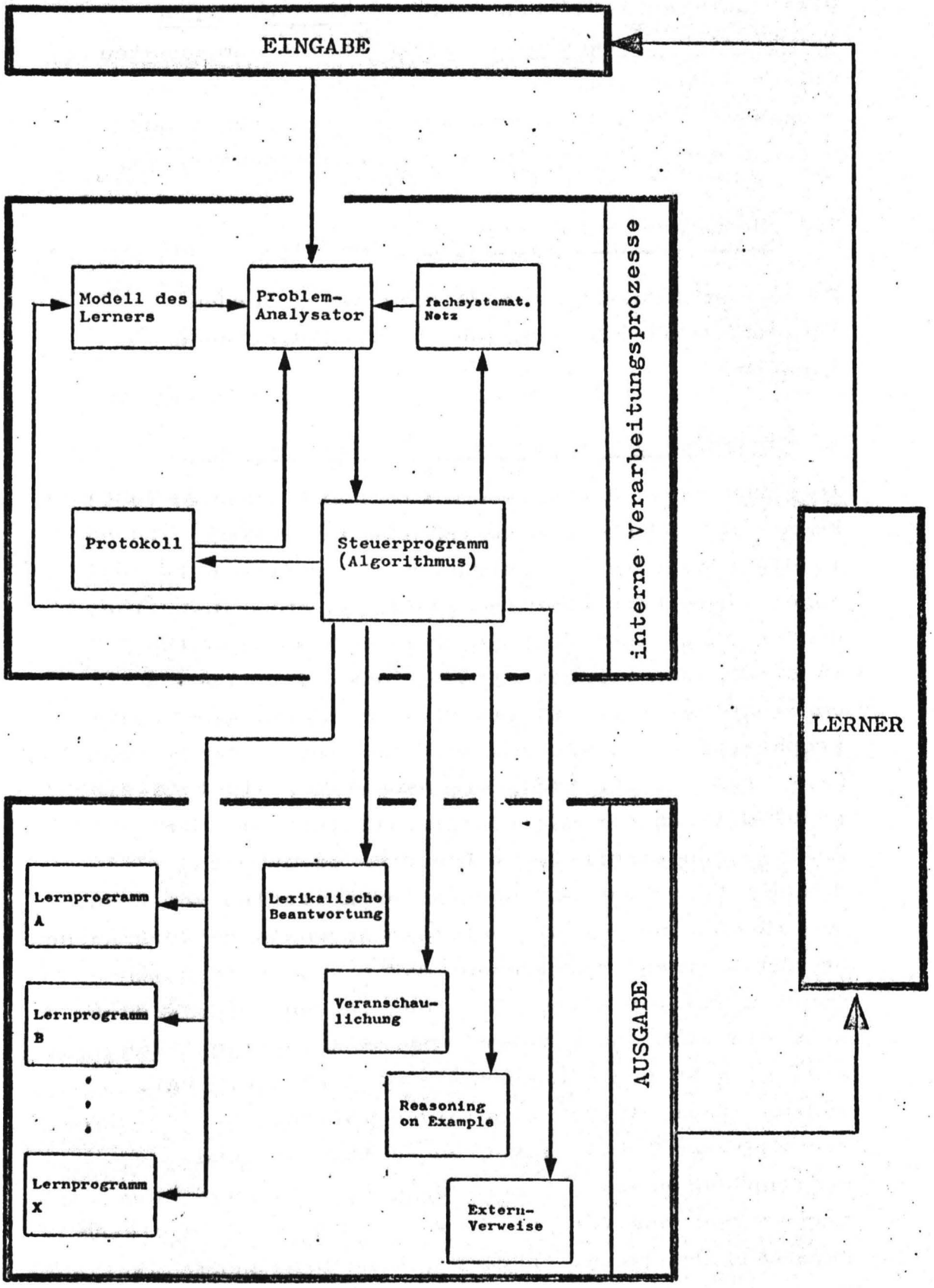


Abb. 2.1 : Regelkreis für ein lerneradaptives Lehrsystem

Diese unterschiedlichen, aber in verschiedenen Lernbereichen anwendbaren funktionalen Komponenten weisen darauf hin, daß ein modularer Aufbau des Programms sowie die Trennung von Algorithmus und Daten sinnvoll erscheint (vgl. auch Abschnitt 4).

2.2 Modularer Aufbau des LALs

Für einen modularen Aufbau sprechen außerdem folgende medienpädagogische bzw. -didaktische Aspekte:

a) Förderung der Eigeninitiative des Nutzers

Im Sinne eines "lebenslangen Lernens" (Dohmen 1976) könnte der Benutzer nach relativ kurzer Zeit in die Lage versetzt werden, sich individuell-abgestimmte Lernbausteine selbst zusammenzustellen. Das Prinzip einer "Lernersteuerung" sollte also in bestimmten Lernsituationen ermöglicht werden, ohne daß damit unreflektiert eine "Lernerkontrolle" propagiert wird, wie sie sich in einigen Untersuchungen (z.B. Judd et al. 1970, Atkinson 1972) als teilweise problematisch herausgestellt hat. Auch bei der Analyse von Lerner-Tutor-Dialogen ergab sich, daß die Lernenden oft nicht über Kriterien verfügen, die ihnen eine für sie optimale Auswahl aus vorhandenen Lernbausteinen ermöglichen, daß sie sich in solch einer Situation zwecks Beratung an den Tutor wenden, wenn sie nicht nach "lernfremden" Kriterien ("Ich sehe so gern bunte Landschaftsbilder" usw./ vgl. Issing et al. 1977) entscheiden wollen.
Trotz dieser Schwierigkeiten sollte aufgrund medienpädagogischer Gesichtspunkte nicht auf eine Lernersteuerung verzichtet werden. Es wird in diesem Zusammenhang darauf ankommen, eine Benutzer-Kommandosprache zu entwickeln, die einerseits leicht erlernbar ist, die andererseits aber die Funktion erfüllt, dem Benutzer Kriterien für sinnvolle Entscheidungen an die Hand zu geben, ohne ihn dabei zu bevormunden.

Um dem Benutzer eine solche Steuerung zu ermöglichen, ist ein modularer Programmaufbau anzustreben, bei dem zwischen Vielfalt und Übersichtlichkeit ein medienpsychologisch sinnvoller Mittelweg gefunden wird und bei dem die Modulbezeichnungen in unterschiedlichen Programmen soweit wie möglich vereinheitlicht werden.

b) Realisierung der Lerner-Adaptivität

Wenn man von einer Systemphilosophie ausgeht, bei der ein menschlicher Tutor bestimmte funktionale Komponenten realisiert, wird man auch aus diesem Grunde durch einen modularen Aufbau möglichst große Übersicht im Programm erreichen müssen. Gerade wenn man berücksichtigt, daß Tutor und Programmautor nicht in jedem Falle identisch sein werden, wird klar, wie schwierig die Aufgabe ist, den Lernzustand bei mehreren Lernenden festzustellen, sich ein Modell von jedem Lerner zu bilden und dementsprechend eine individuell abgestimmte Lernberatung durchzuführen, bei der man eine Beziehung zwischen kognitiver Struktur und fachsystematischem Netz herstellen muß.

c) Modellbildung

Es wäre unökonomisch, würde man für jedes neue Lehrprogramm ein neues "mixed System" (Tutor/Programm) entwerfen. Hier muß man zu Modellen kommen, die es ermöglichen, den Grad der Verallgemeinerung der Ablaufalgorithmen immer weiter zu erhöhen, sodaß neue Programme tatsächlich eine Konkretisierung des entsprechenden Konzepts darstellen. Auch im Sinne dieser Theoriebildung scheint ein modularer, die unterschiedlichen Funktionen analytisch trennender Aufbau angemessen.

d) Programmevaluierung

Ein modularer Aufbau erleichtert es, vorhandene Programme durch die kontrollierte Veränderung von Variablen zu evaluieren. Es ist möglich, Evaluierungsdesigns zu entwerfen, in denen einzelne Module ausgetauscht, weiter differenziert oder "abgehängt" werden. Auch bei neuen fachwissenschaftlichen Erkenntnissen ist dann der Austausch der entsprechenden Informationsblöcke relativ einfach möglich. In Bereichen der Gesellschaftswissenschaften, in denen es häufig verschiedene "Schulen" gibt, wäre es denkbar, daß man unterschiedliche Lehrprogrammteile von unterschiedlichen Autoren erstellen läßt, damit jede Richtung ihre Theorie (Grundlage, Methode, Ergebnisse usw.) in ihrem Sinne optimal darstellen kann.

e) Verzweigung zu anderen Programmen

Die Adaptivität eines Lehrsystems mit Tutorunterstützung könnte den Verweis des Lerners auf andere Programme beinhalten.
Im Lernbereich "Ebbe und Flut" (Issing et al. 1977) kam es häufig vor, daß Lerner die Zusammenhänge und Gesetze von Schwerkraft und Fliehkraft nicht hinreichend genau kannten und an der entsprechenden Stelle auf den Bereich "Mechanik" verwiesen werden mußten. Auf dem Hintergrund einer modular aufgebauten Programmbibliothek sollte man an solch einer Stelle eine tutorunterstützte Verzweigung in ein entsprechendes Lernprogramm vorsehen.

f) Programmnutzung ohne Tutorunterstützung

Bei der Durchführung von LALs mit mehreren Lernern wird sich herausstellen, wie dieses Konzept im ZKTV am besten realisierbar ist.

Ganz sicher wird man nach einem Mittelweg zwischen maximaler Adaptivität und minimalem Aufwand suchen müssen.
So wird man davon ausgehen, bestimmte Teile des Programms ohne Tutorunterstützung ständig anzubieten und das gleiche Programm mit Tutorunterstützung nur zu bestimmten Zeiten.
Auch aus diesem Grunde wäre ein modularer Programmaufbau zu befürworten, damit man je nach Anforderung bzw. Kapazität bestimmte Programm-Module an- oder ab"hängen" kann.

3 Programm VORFAHRT als Beispiel eines LALs

Dieses Programm bezieht sich schwerpunktmäßig auf die Vorfahrtregelung (§8 StVO), die Vorrangregelung beim Abbiegen und an Bahnübergängen (§9, §18 StVO) sowie auf einige andere, damit zusammenhängende Verkehrsregeln (§§ 1, 18, 35, 36, 37, 38 StVO).

3.1 Lernziele

Im Bereich der Bildungstechnologie wird die Notwendigkeit exakter Lernzieldefinitionen mittlerweile nicht mehr diskutiert, sondern fast immer als selbstverständlich vorausgesetzt (vgl. Boeckmann/Lehnert 1975). Diese Lernziele werden - häufig in operationalisierter Form - vom Curriculum- oder Programmautor bzw. vom Lehrer der weiteren Unterrichtsplanung vorangestellt. So sehr diese Methode für "normale" Unterrichtsplanungen sinnvoll sein mag, so problematisch wäre aber ein unreflektiertes Übertragen solcher Verfahren auf ein adaptives Lehrsystem im oben geschilderten Sinne.
Wie ausgeführt, geht es in solch einem System nicht primär darum, Lernende zur Erreichung vorgegebener Lernziele zu bewegen, sondern darum, möglichst weit auf die individuellen Wünsche, Interessen, Möglichkeiten des einzelnen einzugehen. In gewissem Sinne sind damit die Lernziele "offen" (was aber nicht als "laisser-faire-Konzept" zu verstehen ist).

Konkret bedeutet das, daß der Lerner in solch einem System verschiedene Möglichkeiten der Lernzielbestimmung hat.

1) Der Lerner hat kein besonderes Problem, keine spezielle Frage und will auch nicht für eine bestimmte Prüfung (z.B. Führerscheinprüfung) lernen. In diesem Falle wird er etwas im Programm "herumspielen", sich eventuell einige Testaufgaben geben lassen, die er aber eher aus einem spielerischen Verständnis als aus der Notwendigkeit heraus lösen will, die dort abgefragten Informationen in nächster Zeit schnell und exakt abrufen zu müssen. Der Lerner hat für sich quasi kein Lernziel festgelegt, außer dem, daß "es Spaß machen soll". Wenn hier davon die Rede ist, daß kein Lernziel festgelegt wird, so bedeutet das nicht, daß kein Lernprozeß stattfindet. Der Benutzer wird eventuell etwas über seine Beziehung zu dem technischen Medien ZKTV lernen, etwas darüber, wie er sich schnell und unkompliziert in Programmen bewegen kann usw.. Es bedeutet aber, daß nicht zwangsläufig zu einer Überprüfung von Kenntnissen übergegangen wird.

2) Der Lerner hat ein besonderes Problem.
In solch einem Falle wird die Aufgabe des Systems unter anderem darin bestehen, den Lernenden bei der Formulierung des Problems zu unterstützen (Problemanalyse und Problemeinordnung - vgl. auch Abb. 2.1). Das Lernziel wird sich in diesem Falle häufig schon aus der Frage des Lerners ergeben und ist auch oft dann schon erreicht, wenn der Lerner die Beantwortung der Frage als ausreichend empfindet. Die Erfahrung hat jedoch gezeigt, daß sich hinter einer Lernerfrage oft noch andere "Lücken" verbergen. Der Lernende sollte daher auf entsprechende verwandte Probleme hingewiesen werden, auch könnte man ihm eine Selbstüberprüfung durch Testaufgaben nahelegen (vgl. das Unterprogramm EINFUEHRUNG im Programm VORFAHRT).
In der Regel wird der Lerner diese Selbstüberprüfung in Anspruch nehmen, ein benutzerfreundliches System sollte es aber auch erlauben, daß er nach einer für ihn hinreichend erscheinenden Klärung seines Problems das Programm verläßt bzw. zu anderen Programmen verzweigt.

3) Der Lerner hat kein besonderes Problem, muß aber eine Prüfung ablegen, auf die er sich mittels des ZKTVs vorbereiten will. Im Programm VORFAHRT wird das in der Regel die Führerscheinprüfung sein. Wenn ein Lerner seine Schwächen nicht kennt, wird er ein KTV-System dann als benutzerfreundlich empfinden, wenn ihm an dieser Stelle die möglichen - sich aus den Anforderungen der Führerscheinprüfung ergebenden - Lernziele und die entsprechenden Testmöglichkeiten angeboten werden.
Hier zeigt sich konkret, daß eine so verstandene Adaptivität nicht immer "Lernerinitiative" bedeuten muß. In manchen Fällen kann die Adaptivität des Systems gerade darin bestehen, daß der Benutzer die Möglichkeit hat, die Initiative an das System zu delegieren, und dennoch das Eingeben von Wünschen und Zwischenfragen möglich bliebe.

Die angeführten Grobkategorien bei der Behandlung von Lernzielen werden selten in dieser "reinen" Form auftreten. Häufig wird sich herausstellen, daß die Lerner in einem Programmdurchlauf ihre Zielvorstellungen ändern, sich vom System Anregungen geben lassen, um dann in der nächsten Phase wieder relativ "frei" im Programm zu operieren. Gerade diese Möglichkeiten bieten dem Lerner eine große Hilfe für die optimale Ausnutzung seiner Konzentrationsphasen.
So kann man sich beispielsweise vorstellen, daß ein Lerner, der die Führerscheinprüfung machen will, folgendermaßen vorgeht:
Er läßt sich in der ersten Phase die für die Prüfung geforderten Lernziele vorstellen und lernt eine Weile konzentriert (wobei die Initiative hauptsächlich beim LAL liegt).
Eine darauf folgende Phase mangelnder Konzentration überwindet er, indem er eine Weile im Programm "herumspielt".
Hat er diese Phase überwunden, spricht er die individuellen Lernprobleme gezielt an, die ihm erst während der "Spielphase" eingefallen sind.

Das Problem der Lernzieldefinition (ob in operationaler oder anderer Weise) müßte unter diesen Gesichtspunkten neu diskutiert werden, soll ein KTV-System benutzerfreundlich sein. Im Gegensatz zum Schulunterricht, wo er das nur "intern" versuchen kann, kann ein Lerner sich vom KTV-System jederzeit abschalten, wenn ihm die Intentionen des Programms "nicht passen".
Ihm sollte daher ermöglicht werden, daß er entweder gar keine Lernziele (operationalisierte Lernziele im herkömmlichen Sinne) erreichen muß, daß er sich die Lernziele selber setzt und das System ihm Überprüfungsstrategien anbietet oder daß er dem System die Initiative in Hinsicht auf die Lernzieldefinition und -überprüfung übertragen kann.
In bezug auf den letzten Fall wären im Programm VORFAHRT beispielsweise folgende (nicht operationalisierte) Lernziele möglich, die sich in Hinsicht auf Niveau und Zielgruppe stark unterscheiden können:

- Die Vorfahrtregeln kennen
- Die Vorrangregeln beim Abbiegen kennen
- Die Vorrangregeln bei Bahnübergängen kennen
- Die vorfahrtregelnden Verkehrszeichen kennen
- Die Inhalte der Paragraphen der StVO kennen
- Die Vorfahrtregeln anwenden können
- Die Vorrangregeln beim Abbiegen anwenden können
- Die Vorfahrtregeln und Abbiegeregeln in Kombination anwenden können
- Die Vorrangregeln an Bahnübergängen anwenden können
- Die Prüffragen für die theoretische Führerscheinprüfung lösen können
- Häufig auftretende Fehler bei der praktischen Führerscheinprüfung und die Möglichkeiten ihrer Vermeidung kennen.
- Schwierige Vorrangprobleme im Straßenverkehr kennen
- Gerichtsurteile zu Vorrangproblemen kennen

usw.

3.2 Blockdiagramm und Dialogmöglichkeiten

Aus der angesprochenen relativen Offenheit hinsichtlich der Lernziele ergibt sich ein entsprechender Ablaufalgorithmus im Programm VORFAHRT. Er ist gekennzeichnet durch die Möglichkeit des häufigen Hin- und Herspringens zwischen den einzelnen Informationsblöcken, wobei die Ansteuerung dieser Blöcke sowohl vom Lerner als auch vom Tutor ausgehen kann. Im einzelnen ergeben sich folgende Programmteile, die dem Lerner auch im Hauptmenu angeboten werden:

EINFUEHRUNG
TESTAUFGABEN
VORFAHRTREGELNDE VERKEHRSZEICHEN
AUSZUEGE AUS DER STRASSENVERKEHRS-ORDNUNG
FILM

Außerdem wird der Benutzer darauf hingewiesen, wie er sich mit dem Tutor in Verbindung setzen kann:

WENN SIE AN IRGENDEINER STELLE DES PROGRAMMS EINE FRAGE STELLEN WOLLEN ODER EINEN BESONDEREN WUNSCH HABEN, DANN GEBEN SIE BITTE EIN FRAGEZEICHEN EIN.

Im Hauptmenu kann der Lerner das entsprechende Unterprogramm abrufen, indem er die Schreibmarke auf die entsprechende Zeile setzt und dann die Sendetaste drückt oder indem er Kennbuchstaben eingibt (EI, TE usw.). Die Möglichkeit der Verzweigung durch Kennbuchstaben gestattet es ihm, an vielen Stellen aus einem Programmteil gezielt in ein anderes zu springen.

Gibt der Lerner ein Fragezeichen ein, so ist das für den Tutor ein Zeichen, daß der Lerner ein Problem oder eine Frage hat. Er meldet sich dann in der Regel mit "WAS WUENSCHEN SIE ?" (Vom Rechner generierter Text).

Der Tutor verfügt über eine Tutortastatur, die so beschaffen ist, daß er mittels eines Schalters die Schreibinitiative zwischen dem Lerner und sich umschalten kann (Gegenschreiben im Halbduplex). Die Zifferneingabe kann vom Tutor verdeckt vorgenommen werden.
In diesem Falle erscheint die eingegebene Zifernfolge auf keinem Bildschirm, sondern nur auf einer entsprechenden Digitalanzeige beim Tutor.
Der Text "WAS WUENSCHEN SIE?" kann beispielsweise durch die verdeckte Tutoreingabe "1" präsentiert werden.
Hat der Tutor etwas an den Lerner geschrieben, dann schaltet er die Initiative wieder zur Lerner-Tastatur um, damit dieser eventuelle Fragen oder Probleme eingeben kann.
Die Initiativumschaltung könnte in Zukunft auch automatisch, d.h. rechnergesteuert erfolgen.

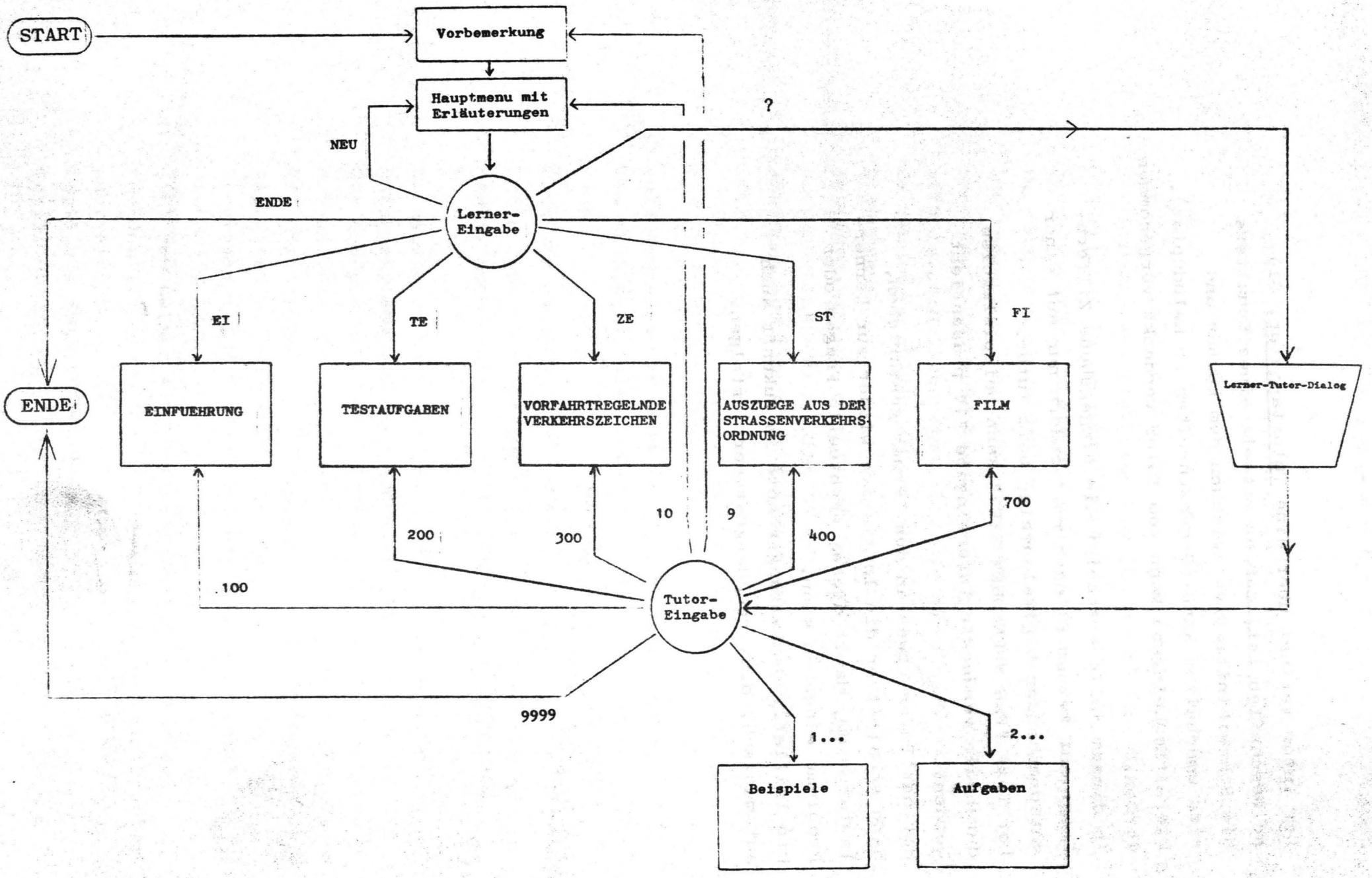


Abb. 3.1 : Blockdiagramm zum Programm VORFAHRT

Die Programmteile VORFAHRTREGELNDE VERKEHRSZEICHEN und AUSZUEGE AUS DER STRASSENVERKEHRS-ORDNUNG sind so strukturiert, daß der Lerner in ihnen (durch Schlüsselworteingabe oder Schreibmarkensteuerung) gezielt zu Untereinheiten zugreifen kann.
Diese Untereinheiten können auch durch entsprechende Schlüsselworteingabe von (fast) jeder anderen Stelle des Programms aus angewählt werden, also auch dann, wenn der Lerner sich nicht gerade mit den Verkehrszeichen oder der StVO beschäftigt!

Mit der Eingabe TE (Tutor: 200) verzweigt das System zu einem Unterprogramm, das der Spezifikation der Lernerwünsche in Hinsicht auf Testaufgaben dient.
Der Lerner kann sich dort beispielsweise entscheiden, ob er gemischte Aufgaben oder Aufgaben zu bestimmten Problemen wünscht und ob diese Aufgaben wiederum in Form von Realsituationen (Dias) oder Graphiken präsentiert werden sollen (vgl. Anhang: A 1).
Die Testaufgaben selbst werden (zur Zeit noch) vom Tutor durch Eingabe der entsprechenden Zahlen zwischen 2201 und 2299 (mit Text) bzw. zwischen 2001 und 2099 (ohne Text) präsentiert. Entsprechendes gilt für die Beispiele.
Die möglichen Präsentationen sind aus dem Anhang ersichtlich.

Die Zifferneingaben sollten in Zukunft durch mnemotechnisch sinnvolle Kurzkommandos ersetzt werden (Notwendigkeit einer Kommandosprache).

Liste der Nutzer- und Tutor-Befehle im Programm VORFAHRT, Tutor-Präsentation von Testaufgaben und Beispielen ausgenommen (führende blanks sind in allen Fällen zugelassen):

Nutzereingabe:	Tutoreingabe:	Reaktion des LALs:
	9	Vorbemerkung zu den im Programm behandelten Verkehrsregeln
NEU	10	Präsentation des Hauptmenus
EI	100	Start einer Einführung in die Vorfahrtregelung, die Vorrangregelung beim Abbiegen und an Bahnübergängen mit der Möglichkeit, sich Beispiele und Testaufgaben zu einzelnen Regeln präsentieren zu lassen.
TE	200	Start eines Unterprogramms zur Auswahl von Testaufgaben (Standbild+Text) über die Vorrangregelung im Straßenverkehr. Der Lerner hat die Möglichkeit, sich gemischte Aufgaben oder Aufgaben zu bestimmten Problemen (Abknickende Vorfahrt, Kreisverkehr usw.) in Form von Graphiken oder Fotos von Realsituationen präsentieren zu lassen.
ZE	300	Präsentation des Menus im Unterprogramm VORFAHRTREGELNDE VERKEHRSZEICHEN
205	205	Präsentation des Zeichens 205 StVO (Vorfahrt gewähren!) mit Erläuterungen
206	206	Präsentation des Zeichens 206 StVO (Halt! Vorfahrt gewähren!) mit Erläuterungen
301	301	Präsentation des Zeichens 301 StVO (Vorfahrt) mit Erläuterungen
306	306	Präsentation des Zeichens 306 StVO (Vorfahrtstraße) mit Erläuterungen
ST	400	Präsentation des Menus im Unterprogramm AUSZUEGE AUS DER STRASSENVERKEHRS-ORDNUNG
P1,P 1	401	Text des § 1 StVO (Grundregeln)
P8,P 8	408	Text des § 8 StVO (Vorfahrt)
P9,P 9	409	Text des § 9 StVO (Abbiegen, Wenden und Rückwärtsfahren)

Nutzereingabe:	Tutoreingabe:	Reaktion des LALs:
P18	418	Text des § 18 StVO (Autobahnen und Kraftfahrstraßen)
P19	419	Text des § 19 StVO (Bahnübergänge)
P35	435	Text des § 35 StVO (Sonderrechte)
P36	436	Text des § 36 StVO (Zeichen und Weisungen der Polizeibeamten)
P37	437	Text des § 37 StVO (Wechsellichtzeichen und Dauerlichtzeichen)
P38	438	Text des § 38 StVO (Blaues Blinklicht und gelbes Blinklicht)
FI	700	Präsentation eines Tonfilms über die Verkehrsregeln, auf die sich das Programm VORFAHRT bezieht. Die Darbietung kann jederzeit durch Drücken der Sendetaste abgebrochen werden. Danach wird das Hauptmenu präsentiert.
?		Einschalten des Tutors. Normalerweise präsentiert das LAL zunächst den Text "WAS WUENSCHEN SIE ?" und wartet auf weitere (frei formulierte) Lernereingaben. Aus diesem "Lerner-Tutor-Dialog" (vgl. Abb. 3.1) kann nur durch Tutoreingabe zu anderen Programmteilen verzweigt werden. Der Lerner hat lediglich die Möglichkeit, durch ENDE oder NEU zu verzweigen.
ENDE	9999	Ende des Programms. Dem Lerner wird das Menu aus dem Bereich "Lernen im Dialog" präsentiert.
	1	Präsentation des Textes WAS WUENSCHEN SIE ? mit anschließendem carriage return

Die Angaben über die Tutor-Befehle beziehen sich auf die verdeckte Eingabe. Bei nicht-verdeckter Eingabe müssen eine schließende eckige Klammer und vier Ziffern eingegeben werden (also]0200 statt 200,]0009 statt 9 usw.).

Will der Lerner sich nicht über die Tastatur mit dem Tutor verständigen, sei es, daß es aufgrund der fehlenden Übung zu langsam geht oder daß sein Problem so kompliziert ist, daß es nicht in wenigen Worten dargestellt werden kann, so kann auf Anforderung des Lerners (Eingabe: SPRECHEN) vom Tutor ein akustischer Kanal geschaltet werden, auf dem sich dann der "Lerner-Tutor-Dialog" vollzieht.
Obwohl durch den akustischen Kanal die Lerner-Probleme normalerweise kurz und präzise (u.a. durch häufige kurze Rückfragen, Umschreibungen usw.) analysiert und eingeordnet werden können, sollte auf den alphanumerischen Dialog als "Ausgangsbasis" nicht verzichtet werden.
Dafür gibt es mehrere Gründe:

1) Für einige Lerner kann es förderlich sein, wenn sie sich die Formulierung ihrer Frage oder ihres Problems vor der Eingabe genau überlegen müssen und nicht einfach "drauflossprechen" können. Oft wird sich herausstellen, daß die dazu notwendigen Prozesse beim Lerner dazu führen, daß er auch ohne Anrufung des Tutors seine Probleme löst, indem er z.B. selbst zu bestimmten Informationsbausteinen (im Falle des Programms VORFAHRT etwa zu den entsprechenden Verkehrszeichen oder Gesetzestexten) zugreift. Hierdurch kann u.a. der Aufbau von Lernstrategien (Selbstinstruktionsstrategien) beim Lerner unterstützt werden.

2) Im Sinne einer Modellbildung für interaktive Dialogdienste muß ständig überprüft werden, welche Nutzeranforderungen vom Rechner verstanden und beantwortet werden können. Hierfür können die (aufzuzeichnenden) alphanumerischen Dialoge die empirische Grundlage sein.

3) Der Nutzer wird allgemein mit der Handhabung der Tastatur vertrauter (Übungseffekt).

3.3 Lernmaterial

Das Lernmaterial des Programms VORFAHRT soll an dieser Stelle nur kurz umrissen werden, um dem Leser beispielsweise das Verständnis des Dialogprotokolls (Abschnitt 3.4.) zu erleichtern. Eine Auflistung der Standbilder befindet sich im Anhang (A2).

Rechnerschrift

In Form von Rechnerschrift (nur Großbuchstaben, keine Umlaute) sind vorhanden:

- das Hauptmenu mit Erläuterungen
- der Text der Einführung mit Beispielerläuterungen
- die Erläuterungen über die Auswahl von Testaufgaben und die Fragen zu den Testaufgaben sowie die Bestätigung der richtigen Lösung
- die Erläuterungen zu den vorfahrtregelnden Zeichen (Zeichen 205, 206, 301, 306 StVO)
- die Gesetzestexte der §§ 1, 8, 18, 19, 35, 36, 37, 38 StVO
- Erläuterungen zur Präsentation des Films
- nur vom Tutor abrufbar: WAS WUENSCHEN SIE ? (Eingabe: 1)

Außerdem steht dem Tutor die Möglichkeit zur Verfügung, während des Lerner-Tutor-Dialogs freien Text über die Tutortastatur einzugeben bzw. auf diesem Wege den gerade präsentierten Text zu modifizieren.

Standbild

Als Standbild stehen 73 Dias zur Verfügung (s. A 2). Bei den meisten von ihnen handelt es sich um reale oder graphische Darstellungen von Verkehrssituationen oder Verkehrszeichen. Zwei der Dias dienen der farbigen Hinterlegung von Rechnerschrift.

Bewegtbild

Als Bewegtbild steht der Film "Nach Ihnen - bitte" (Verkehrs - Verlag Remagen) in Form einer Kopie auf VCR-Cassette zur Verfügung. Das Bewegtbild kann (im Gegensatz zum Standbild) z.Z. aus technischen Gründen nicht mit Rechnerschrift überlagert werden.

Ton

Ton existiert im Programm nur im Zusammenhang mit dem Film "Nach Ihnen - bitte". An jeder Stelle des Lerndurchlaufs kann aber ein akustischer Kanal vom Tutor geschaltet werden. Der Ton kann der Rechnerschrift, dem Standbild und dem Bewegtbild überlagert werden.

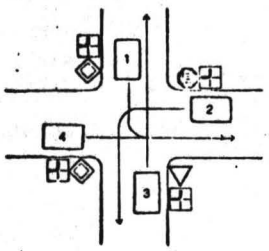
3.4 Dialogprotokoll

Das folgende Dialogprotokoll ist teilweise ein hypothetisches. Es setzt sich aus Dialogsequenzen mehrerer Lerner zusammen, die einerseits durch die Arbeit der Projektgruppe "Individuelles interaktives Lernen" (Issing et al. 1977) in Form von Protokollen vorliegen, andererseits von Mühlbach aufgrund seiner Erfahrungen als Tutor während der Nutzung des Programms VORFAHRT auf der Berliner Funkausstellung (vgl. Abschnitt 5) erstellt wurden. Aus Gründen der Veranschaulichung interessanter Interaktionsstrukturen werden diese Sequenzen als der Lerndurchlauf eines (hypothetischen) Lerners dargestellt.

Lernereingabe	Tutoreingabe	Präsentation	Bemerkungen
			Lerner wählt das Programm VORFAHRT an.
		Vorbemerkung	
(Leerzeile)			
		Hauptmenu	Der Lerner ist durch Eingabe einer Leerzeile zum Hauptmenu gekommen, in dem er auf die Möglichkeit einer Fragestellung hingewiesen wird.
?		?	"Lerner-Tutor-Dialog"
	1		verdeckte Tutoreingabe
		WAS WUENSCHEN SIE ?	Es muß noch untersucht werden, ob die Präsentation dieses Textes in jedem Falle optimal ist, deshalb die Präsentation erst nach Tutoreingabe.
HAT IMMER DER VORFAHRT, DER VON RECHTS KOMMT ?		HAT IMMER DER VORFAHRT, DER VON RECHTS KOMMT ?	Der Lerner kann seine Eingabe auf dem Bildschirm kontrollieren
	408		verdeckte Tutoreingabe
		Gesetzestext des § 8 StVO (Vorfahrt) Darin werden u.a. die vorfahrtregelnden Zeichen erwähnt.	
? BITTE INFORMATIONEN ZU DEN ZEICHEN		? BITTE INFORMATIONEN ZU DEN ZEICHEN	Der Tutor wird angerufen (?) und gleichzeitig die Anforderung eingegeben.
	300		
		Menu des Unterprogramms VORFAHRTREGELNDE VERKEHRSZEICHEN (Standbild mit den Zeichen 205, 206, 301, 306 StVO und Überlagerung Fachnerschrift)	

Lernereingabe	Tutoreingabe	Präsentation	Bemerkungen
306			Der Lerner wählt aus dem Menu aus, indem er den Cursor auf die entsprechende Zeile setzt und die Sendetaste drückt.
		Zeichen 306 ("Vorfahrt-straße") mit Erläuterungen (Standbild + Rechnerschrift)	
(Leerzeile)			Der Text wird in drei Blöcken präsentiert. Nach Eingabe einer Leerzeile erscheint jeweils der nächste Block. Nachdem der Lerner alle Erläuterungen zum Zeichen gelesen hat, gibt er wieder eine Leerzeile ein.
(Leerzeile)			
(Leerzeile)			
		Programmbereichs-anzeige + MOECHTEN SIE NOCH ERLAEUTERUNGEN ZU ANDEREN VORFAHRT-REGELNDEN ZEICHEN? JA NEIN	
NEIN			Eingabe durch Cursor-positionierung
		Hauptmenu	
EI			Eingabe durch Cursor-positionierung
		Start des Unter-programms EINFUEHRUNG	
(Leerzeile)		.	Das "Blättern" erfolgt jeweils durch Eingabe einer Leerzeile
(Leerzeile)		.	
.			
.			
.			

Lernereingabe	Tutoreingabe	Präsentation	Bemerkungen
TE		TE	Der Lerner hat sich die Kennbuchstaben für das Unterprogramm TESTAUFGABEN gemerkt und verzweigt selbst dorthin
		Start des Unterprogramms TESTAUFGABEN (Vorbemerkung)	
?		?	Der Lerner hat ein spezielles Problem.
	1		
		WAS WUENSCHEN SIE ?	
BITTE EINE AUFGABE ZUR ABKNICKENDEN VORFAHRT		BITTE EINE AUFGABE ZUR ABKNICKENDEN VORFAHRT	
	HABEN SIE DA EIN BESONDERES PROBLEM ?	HABEN SIE DA EIN BESONDERES PROBLEM ?	Der Tutor hat die Schreibinitiative auf seine Tastatur geschaltet und muß sie, wenn er fertig ist, zum Lerner zurückschalten.
SCHWIERIG ZU BESCHREIBEN!		SCHWIERIG ZU BESCHREIBEN!	
			Der Tutor schaltet den akustischen Kanal zum Lerner und unterhält sich dann mit ihm über sein Lernproblem. Dabei stellt sich heraus, daß der Lerner die Situation nicht hinreichend strukturieren kann, wenn aus verschiedenen Richtungen Fahrzeuge kommen.

Lernereingabe	Tutoreingabe	Präsentation	Bemerkungen
	2261		verdeckte Eingabe
		Testaufgabe zur abknickenden Vorfahrt (Dia 61 mit Text) IN WELCHER REIHENFOLGE DARF GEFAHREN WERDEN ? 	
4 1 3 2		4 1 3 2	
	LEIDER FALSCH	LEIDER FALSCH	Umschaltung der Initiative
1 4 3 2		1 4 3 2	
	WIEDER FALSCH. MOECHTEN SIE LOESUNGSHILFEN?	WIEDER FALSCH. MOECHTEN SIE LOESUNGSHILFEN?	Die Antwortanalyse erfolgt noch vom Tutor
JA		JA	
	112		verdeckte Eingabe! Der Tutor verzweigt direkt zu einem Informationsblock innerhalb des Unterprogramms EINFUEHRUNG.
		Regel für gleichrangige Straßen bei abknickender Vorfahrt	
	]2261	]2261	offene Eingabe! Der Tutor schreibt jetzt den Code für die bearbeitete Testaufgabe in die Zeile, wo die Schreibmarke steht. Für den Lerner bedeutet das Drücken der Sendetaste "WEITER" - da der Code abgeschickt wird, kommt er zu der zu lösenden Testaufgabe!
]2261			"Eingabe" durch Drücken der Sendetaste

Lernereingabe	Tutoreingabe	Präsentation	Bemerkungen
		Testaufgabe zur abknickenden Vorfahrt (s.o.)	
4 1 2 3		4 1 2 3	Lösung wird vom Tutor analysiert
	297		verdeckte Eingabe
		Programmbereichs-anzeige u. Text: SIE HABEN DIE AUFGABE RICHTIG GELOEST! MOECHTEN SIE NOCH EINE TESTAUFGABE? JA NEIN ICH MOECHTE NEU WAEHLEN	
ICH MOECHTE DEN FILM SEHEN		ICH MOECHTE DEN FILM SEHEN	Die Eingabe kann vom Rechner an dieser Stelle noch nicht interpretiert werden! Er erwartet JA, NEIN oder ICH MOECHTE NEU WAEHLEN. Er sendet deshalb eine Leerzeile und wartet auf eine neue Eingabe - keine Fehlermeldung! Der Tutor schaltet sich ein und verzweigt direkt zum Film. - Der Lerner fühlt sich verstanden!
	700		verdeckte Eingabe!
		Start des Unterprogramms FILM	
(Leerzeile)			Durch Drücken der Sendetaste wird die Präsentation des Films abgebrochen und es erscheint das Hauptmenu.
		Hauptmenu	
ENDE		ENDE	Der Lerner beendet den Lerndialog

4 Implementierung

Nach dem vorliegenden Konzept für ein Lerner-adaptives Lehrsystem wurde das Programmsystem "Vorfahrt" als ein Demonstrationsbeispiel des HHI Laborprojekts "Zweiwegkabelfernsehen" zur Internationalen Funkausstellung 1977 implementiert.

4.1 Allgemeines

4.1.1 Strukturformen für eine TKF (1)

Ausgehend von der Baumstruktur kann eine lineare, hierarchische Menustruktur (LHM) definiert werden. Für eine solche Struktur gibt es folgende Darstellungsmöglichkeit:

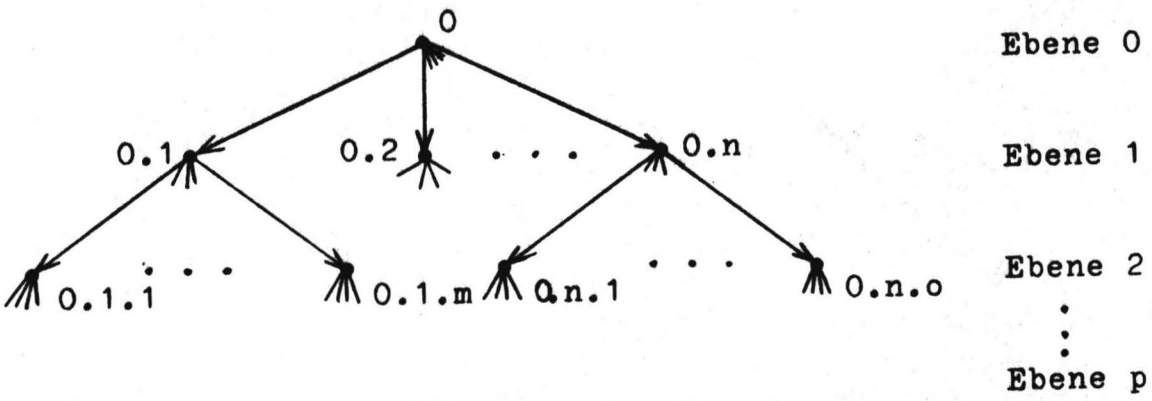


Abb.4.1 Lineare, hierarchische Struktur. (Die Ebene ergibt sich aus der Anzahl der Punkte zwischen den Ziffern)

Ein Beispiel hierzu möge folgendermaßen aussehen:

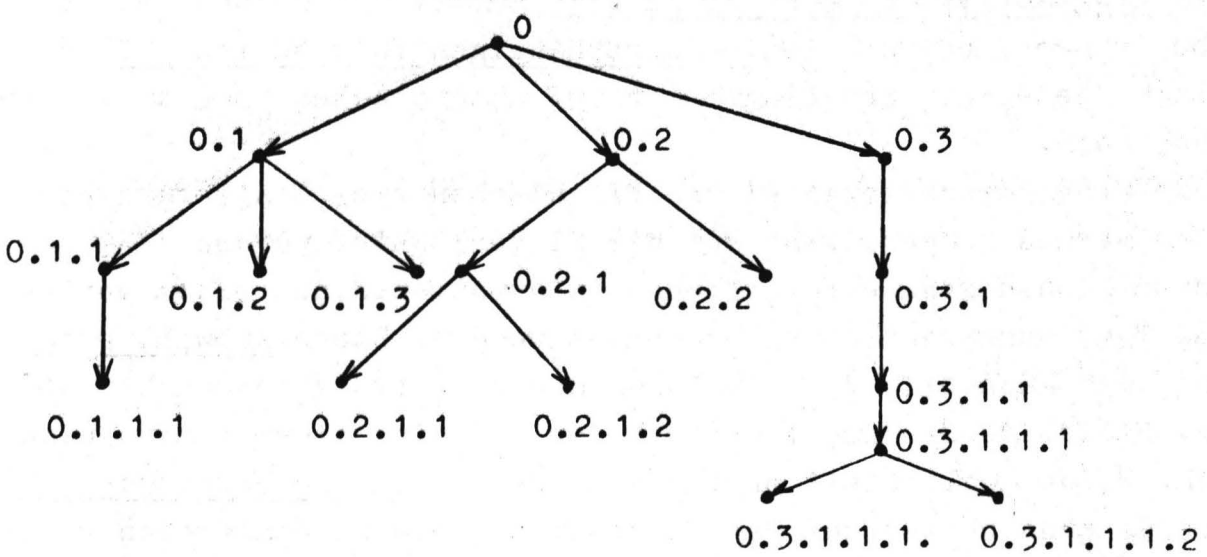


Abb.4.2 Beispiel zur LHM

In dieser Strukturform sind keine Quersprünge möglich. Will man z.B. vom Knoten 0.1 zum Knoten 0.2.1.2,muß man zunächst zum Knoten 0 zurück und von dort den 0.2-er Ast Knoten nach Knoten abwärts laufen bis zum gewünschten Knoten. Dies vermeidet man durch eine zirkuläre, adaptive Menustruktur (ZAM). In dieser Struktur können Äste nicht nur abwärts sondern auch aufwärts verfolgt werden. Außerdem bietet sie die wichtige Möglichkeit, Quersprünge vornehmen zu können. Nachfolgend wird ein Ausschnitt der möglichen Bewegungen von und nach Knoten 0.1. dargestellt:

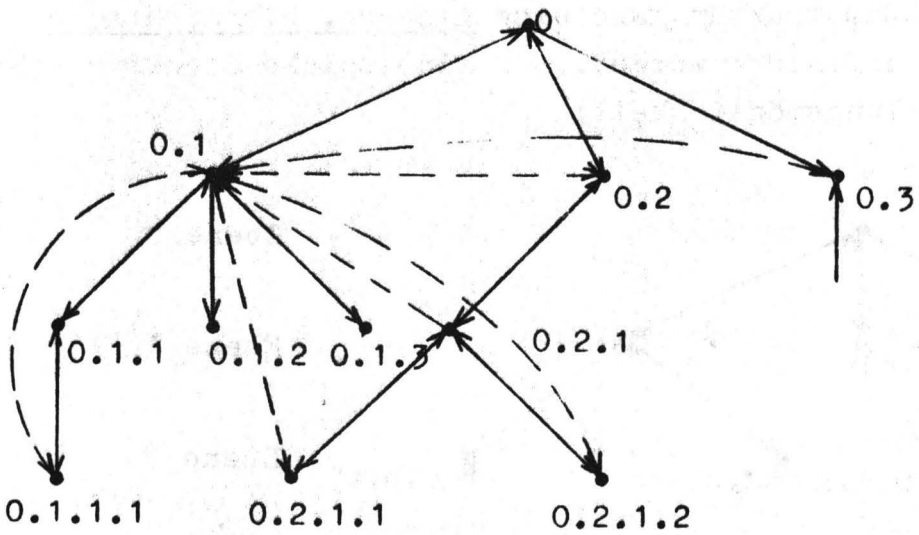


Abb. 4.3 Beispiel zur ZAM ——— lineare Äste
-------- zirkulare Maschen

4.1.2 Eingabeformen für eine TKF (1)

Es ist möglich, dem Nutzer zwei Eingabeformen anzubieten:
Bei der programmierten Interaktion (PI) wartet der Rechner auf eine Eingabe, während bei der teilnehmerabhängigen Interaktion (TI) der Teilnehmer jederzeit den Programmablauf unterbrechen kann und Daten absenden kann.
Wenn die Programmschritte bei der PI zwischen zwei Eingaben klein gehalten werden können, kann auf die TI verzichtet werden.
Als Daten können gerade zutreffende Informationen aus einem vorgegebenen Text ausgewählt und abgesendet werden. Diese Auswahlmethode (AM) hat den Nachteil, daß nur Daten aus dem Text abgeschickt werden können. Beliebige Sprünge innerhalb der TKF sind hiermit also nicht möglich. Diese Möglichkeit ergibt sich bei der Freiwahlmethode (FM). Bei dieser kann beliebiger Text über eine Tastatur eingegeben werden. Allerdings wird damit auch das Vorhandensein einer solchen Tastatur notwendig und die zur Auswertung des Textes benötigte Rechenzeit ist

größer als die bei der Auswahlmethode.

Aus der Erkenntnis, daß ein Lerner-Tutor-Dialog (LTD) dem Lernenden oder auch Teilnehmer ein besonders effektives Lernen ermöglicht, ergeben sich zwei Typen von Nutzern: Der Teilnehmer (Tn) und der Tutor (Tt). Eingaben können von beiden vorgenommen werden. Damit das Programm unterscheiden kann, welcher Nutzertyp die Eingaben vorgenommen hat, muß den abgesandten Daten ein entsprechendes Steuersignal mitgegeben werden.

Um im LTD die Initiative möglichst beim Tt zu lassen, ist es sinnvoll, die Tn-Eingabemöglichkeiten einzuschränken.

---

(1) siehe auch: P. Mahnkopf, Kurzer Statusbericht zu den Telekommunikationsformen des HHI-KTV-Systems, Stand Mai 1977

4.2 Programmsystem "Vorfahrt"

Das Programmsystem "Vorfahrt" hat eine ZAM, bei der allerdings nicht alle Knoten von beliebigen Knoten aus angesprungen werden können. Es gibt Knoten, die vom Tn und Tt angesprungen werden können, und solche, die nur vom Tt,und andere, die nur linear erreicht werden können. Knoten sind hier Textausgaben mit anschließendem Warten auf Eingabe.

Auf dem Bildschirm steht z.B.:

Möchten Sie ein Beispiel sehen?
Ja
Nein

Das Programm wartet auf eine Eingabe.

Abb. 4.4 Beispiel für einen Knoten

Ein Knoten entspricht meist einer Bildschirmseite.

Es gibt 4 Arten von Knoten:

1) • Knoten, der nur in linearem, hierarchischem Ablauf nach Absenden einer Leerzeile oder eines anderen Textes bei vorherigen Verzweigungen vom vorherigen Knoten erreicht werden kann.

2) ● n Knoten, der außerdem noch vom Tt über Eingabe der Zahl n angesprungen werden kann.

3) xT,n Knoten, der außerdem noch vom Tn über Eingabe von Text T angesprungen werden kann.

4) ⊙ LTD-Knoten, der vom Tn nur durch wenige erlaubte Befehle verlassen werden kann.

Steht eine eingeklammerte Zahl neben dem Knoten, heißt dies, daß der Knoten unter dieser Zahl nicht anspringbar ist (sie entspricht dem Statement-Label im Programm).

Zum Abschicken von Text stehen die AM und die FM zur Verfügung. Nach Absenden eines Textes werden die ersten vier Zeichen, die in IREAD (1) stehen, untersucht, ob sie einem Knoten zugeordnet werden können. Falls dies nicht der Fall ist, wird eine Leerzeile erzeugt und eine neue Eingabe erwartet. Handelt es sich um eine Tutoreingabe, sind diese vier Zeichen die Zeichen #$#$. In IREAD (2) steht dann die Zahl, die der Tutor eingegeben hat. Kann diese keinem Knoten zugeordnet werden, wird wiederum eine Leerzeile erzeugt und auf Eingabe gewartet.
Das Absenden einer Leerzeile (oder eines "Verzweigungstextes") bedeutet, daß im linear hierarchischem Ablauf fortgefahren werden soll. Dieser Ablauf ist unterbrochen, wenn der LTD erreicht ist. In diesem Fall kann bis auf Ausnahmen nur noch der Tutor Eingaben vornehmen. Solche Ausnahmen sind die Eingaben "NEU" und "ENDE".

Die Struktur des Programmes selber ist zunächst hierarchisch. Die ZAM wird dadurch erreicht, daß Verteilerknoten existieren (VK), die die gewünschten Sprünge organisieren. Wird nun ein Text abgeschickt, der nicht dem oder den hierarchisch nachfolgenden Knoten entspricht, wird zunächst untersucht, ob der Text überhaupt in einer der Textlisten steht. Steht er dort nicht, wird eine Leerzeile erzeugt und auf eine neue Eingabe gewartet. Ist der Text jedoch in der Liste, wird der VK im Hauptprogramm (VKØ) angesprungen, von dem aus der gewünschte Sprung organisiert wird. Entsprechendes geschieht bei der Tutoreingabe.

Nachstehend ein Beispiel zu einem Sprung:

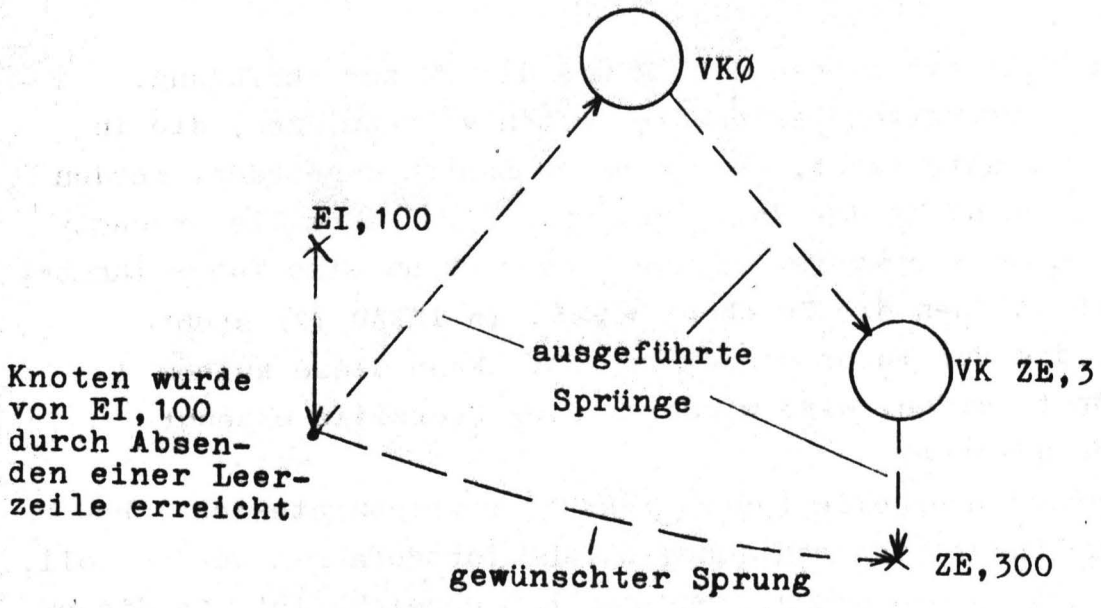


Abb. 4.5 Ausführung eines Sprung

Der Verteiler VKØ wird dabei stets angesprungen. Ob noch ein weiterer Verteiler notwendig wird (hier VK ZE, 3) ergibt sich daraus, ob im Unterprogramm weitere Sprünge notwendig sind.
Anhand eines Ausschnittes aus einem Flußdiagramm mit LTD soll in einem weiteren Beispiel gezeigt werden, wie das Programm von einem Knoten zum nächsten gelangt.

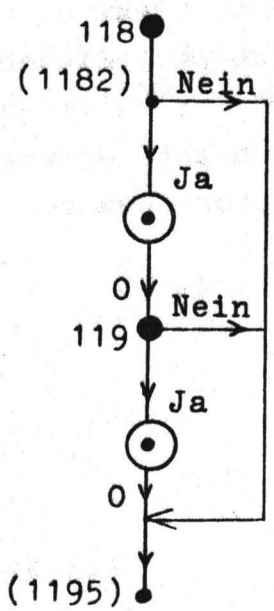


Abb. 4.6 Flußdiagrammausschnitt mit LTD

Der Tn ist im linearen Abstieg oder durch den Ansprung durch den Tt nach Knoten 118 gelangt. Nach dem Absenden einer Leerzeile gelangt er zum Knoten (1182). Hier soll er sich entscheiden, ob er ein Beispiel sehen will. Entscheider er sich für "NEIN", gelangt er direkt zum Knoten (1195). Bei "JA" läuft er auf einen Knoten des LTD's auf. Solange er sich auf diesem Knoten befindet, können Tn und Tt über den Bildschirm miteinander kommunizieren (Duplex-Betrieb).
Der Knoten wird nur dann verlassen. wenn der Tn einen der noch erlaubten Befehle abschickt oder der Tt einen entsprechenden Sprung vornimmt. Der Tt kann nun entscheiden, ob er dem Tn ein zu dem Lehrstoff gehörendes Beispiel zeigen will, oder ob er den Tn zu einem anderen Programmteil hin verzweigen will.

Falls er dem Tn ein entsprechendes Beispiel gezeigt hat, erreicht der Tutor durch Absenden von "119" oder einer "Ø" zu dem Knoten 119 - (letzteres gedacht als Erleichterung für den Tt). Falls der Tt in andere Programmteile verzweigt hat, gelangt er nur durch Absenden von "119" zu diesem Knoten. An diesem Knoten wird der Tn gefragt, ob er noch ein weiteres Beispiel sehen will. Entscheidet er sich jetzt für "NEIN", gelangt er wiederum direkt zum Knoten (1195). Mit "JA" läuft er wieder auf einen LTD-Knoten auf. Dieser Knoten kann wiederum vom Tn nur über einen der noch erlaubten Befehle verlassen werden. Der Tutor erreicht den Knoten (1195) nach Zeigen eines entsprechenden Beispiels durch Abschicken einer "Ø".
Von jedem dieser Knoten kann der Tt jeden von ihm anspringbaren anderen Knoten im Programm erreichen. Das gleich gilt für den Tn, solange es sich nicht um einen LTD-Knoten handelt. Diese Knoten kann der Tn nur über wenige erlaubte Befehle verlassen.(Implementiert: "NEU" für Hauptmenu und "ENDE" zum Beenden). An jedem Knoten kann auch beliebiger Text abgesandt werden. Solange der Anfang dieses Textes keinem erlaubten Befehl entspricht, wird der Knoten nicht verlassen. Tn und Tt können also an jedem Knoten unter der genannten Einschränkung miteinander kommunizieren.

Im nachfolgenden ist das übergreifende Hauptprogramm (HP) als Übersichtsdiagramm dargestellt:

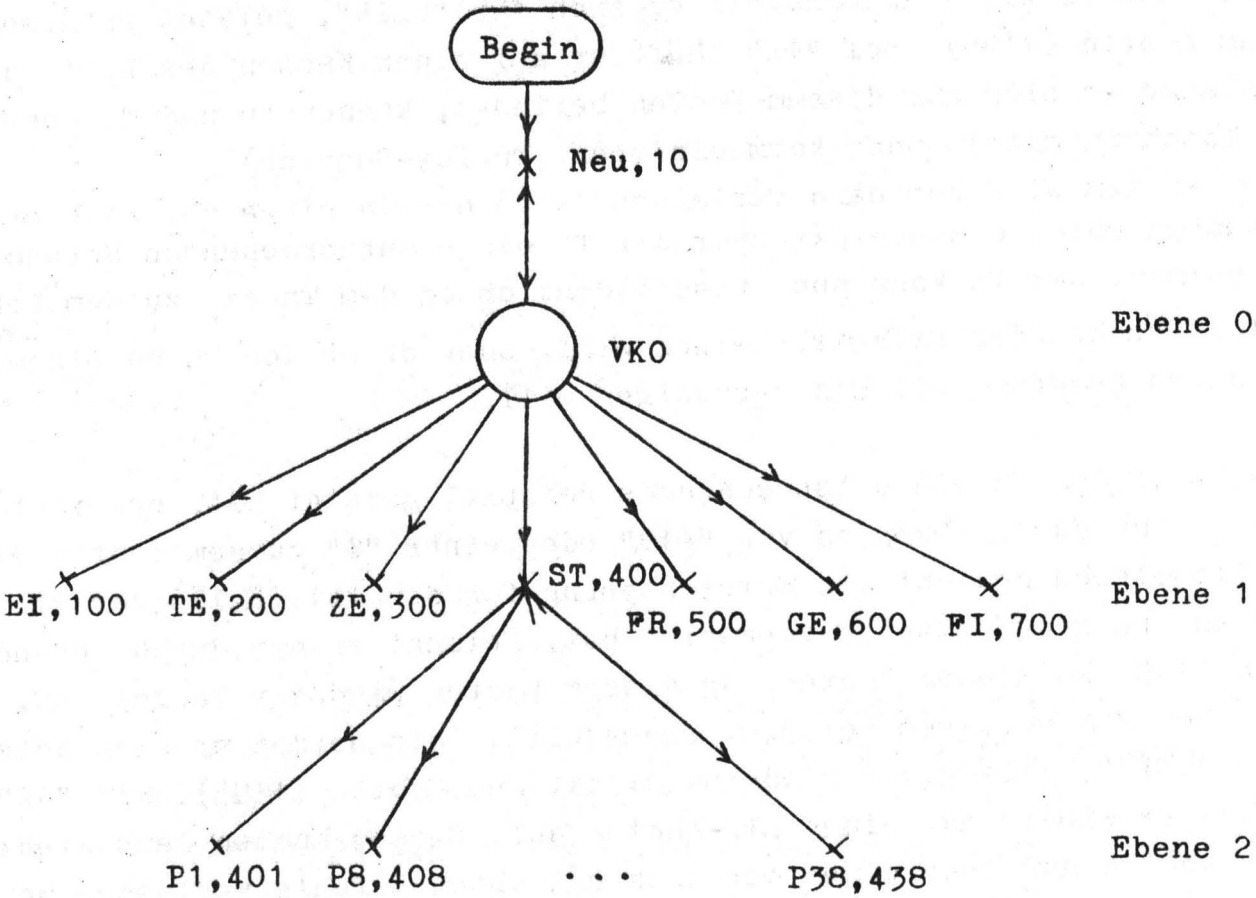


Abb. 4.7 Hauptprogramm

Der Verteilerknoten verteilt zu allen dargestellten Knoten. Jeder Knoten entspricht dem Aufruf eines Unterprogramm (UP), in dem - falls erforderlich - weiter verteilt wird. Niedrigere Ebenen als die Ebene 1 werden normalerweise in den Unterprogrammen behandelt. Nur die Ebene 2 zu den Knoten ST,400 befindet sich im HP. (Der Grund hierfür liegt darin, daß ein bereits vorhandenes Programm erweitert worden ist, diese Ebene 2 also bereits im HP programmiert war.) Der Teilnehmer erreicht z.B. den Knoten ZE,300 durch Absenden von " ZE " oder "ZE ". Entsprechendes gilt für alle anderen vom Tn durch Sprung erreichbaren Knoten. Es müssen also stets mindestens vier alphanumerische Zeichen abgeschickt werden.
Der Tutor seinerseits erreicht den Knoten ZE,300 durch Absenden von "]0300" (vorgesehen ist auch "]300"). Das Zeichen "]" wird dabei in"#$#$" umgewandelt und dient dem Programm als Erkennungszeichen

dafür, daß eine Tutoreingabe vorliegt.(Falls eine spezielle Tutoreingabe möglich ist, kann der Tt auf Eingabe des Zeichen "]" verzichten). Entsprechendes gilt für alle anderen durch Tt-Befehle erreichbaren Knoten.
Der Tn kann Text nach der AM oder nach der FM absenden. Bei der AM setzt er eine blinkende Marke auf die gewünschte Zeile und schickt sie ab. Bei FM schickt er selbst geschriebenen Text ab und kann damit im Gegensatz zur AM auch "freie" Sprünge vornehmen.

Nun sind natürlich nicht nur die dargestellten Sprünge möglich. Es gibt weitere Sprünge, die direkt auf eine niedrige Ebene in einem UP springen. Zum Beispiel wird mit den assoziativem Sprung "306" in das gleich UP gesprungen wie mit "ZE ", jedoch an eine andere Stell des UP's. Entsprechendes gilt für den Tutorsprung. Mit "]219" folgt ein Sprung in das gleich UP wie mit "J200", jedoch an eine andere Stelle. Die Hunderterstelle gibt hierbei das UP an, in das gesprungen wird. Ausnahmen hiervon sind "205" und "206", die in das UP vom 300 führen. Dies deshalb, weil diese Zahlen mit den Nummern von Verkehrszeichen übereinstimmen und so als mnemotechnische Hilfe für den Tutor dienen können.

Weitere Einzelheiten zum Programmsystem sind im Anhang aufgeführt.

5 Erfahrungen bei der Programmnutzung auf der Funkausstellung

Das Programmsystem VORFAHRT lief (in der Fassung vom 25. 8. 1977) im Rahmen der Demonstration des HHI-Laborprojekts "Zweiweg-Kabelfernsehen" auf der Internationalen Funkausstellung (26. 8. - 4. 9. 1977). Die dabei gemachten Erfahrungen hinsichtlich der Programmnutzung sollen hier kurz dargestellt werden.

5.1 Erfahrungsgrundlage

Als Erfahrungsgrundlage dienten die Lerndialoge und einige Gespräche mit den Besuchern des HHI-Standes. Diese Stichprobe ist weder in Hinsicht auf soziostrukturelle Kriterien noch in Hinsicht auf Nutzer-Motivation eines ZKTV-Systems als repräsentativ zu betrachten.
Verallgemeinerungen der auf der Funkausstellung gemachten Erfahrungen sind deshalb nur sehr bedingt zulässig.
Im Sinne einer formativen Systemevaluierung ist aber die Darstellung der erfahrenen Schwierigkeiten und Möglichkeiten bei der konkreten Dialogdurchführung durchaus nützlich (vgl. Issing 1976).

5.2 Anrufen des Tutors

Das Programmsystem VORFAHRT diente auf der Funkausstellung primär der Demonstration eines Dienstes mit Tutorunterstützung und erst sekundär einer verkehrspädagogischen Zielsetzung. Das ergab sich u.a. daraus, daß die Besucher in der Regel weder Zeit noch Lust hatten, sich über "verzwickte" Vorrangregeln zu informieren. Sie wollten in erster Linie diese Art der Telekommunikation ausprobieren. Das führte dazu, daß Fragen an den Tutor oft nicht "echt" waren, sondern lediglich gestellt wurden, um zu überprüfen, ob das System "funktioniert".

Rückschlüsse auf die Programmstruktur aus didaktischer Sicht sind auf dieser Basis schwer möglich.
Die Fragen, die gestellt wurden, waren auch nicht immer "ernsthaft". Oft wurde versucht, den Tutor irgendwie (in einem scherzhaften Sinne) "hereinzulegen" (z.B.: WENN DER MOTORRADFAHRER BETRUNKEN IST, HAT DER ANDERE VORFAHRT).

Auch entwickelte sich der Lerner-Tutor-Dialog aufgrund dieser Umstände nicht immer in die Richtung, die in einem LAL angestrebt wird.
Er diente oft nicht Funktionen wie Problemanalyse und -einordnung mit anschließender Verzweigung und Übernahme der Initiative durch den Lerner, sondern erschöpfte sich häufig in einer lockeren, scherzhaften "alphanumerischen Konversation".
Ob diese Entwicklung von Lerner-Tutor-Dialogen nur Ausdruck der Umstände auf der Funkausstellung war oder eine generelle Gefahr bei diesem Dienst ist, sollte auf jeden Fall untersucht werden.

5.2.1 Alphanumerischer Dialog

Beim alphanumerischen Dialog traten nach kurzen Erläuterungen zur Tastatur (Schreibmarkensteuerung, Sendetaste usw.) keine nennenswerten Schwierigkeiten auf.
Die Nutzer waren anscheinend mit einer Schreibmaschinentastatur soweit vertraut, daß sich die Eingabe auch mit einem "Zweifinger-Suchsystem" als benutzerfreundlich darstellte.
Von keinem Besucher wurde die Anordnung der alphanumerischen Zeichen generell kritisiert, einige Besucher waren angenehm überrascht, daß sie ihre Schreibmaschinenkenntnisse in einem anderen als dem gewohnten Zusammenhang anwenden konnten, was die Attraktivität des Mediums ZKTV für sie sichtbar erhöhte.

> Man kann also sagen, daß die Nutzer-Tastatur sich in dem hier angesprochenen Zusammenhang im großen und ganzen bewährt hat und daß deshalb auf dieser Grundlage weitergedacht werden sollte.

Kritisiert wurde allerdings die (im Unterschied zur deutschen Schreibmaschinentastatur) andere Lage von Y und Z und die fehlende Wiederhol- bzw. Dauer-Funktion bei der Schreibmarkensteuerung.

Auch die Tutor-Tastatur hat sich von ihrer Konzeption her im großen und ganzen bewährt. Allerdings konnten zum Zeitpunkt der Funkausstellung (aus technischen Gründen) noch keine Erfahrungen mit der verdeckten Tutoreingabe gemacht werden. Das Erscheinen der Code-Nummern des Tutors auf dem Nutzer-Bildschirm führte zwar gelegentlich zu Erstaunen, wurde aber auch nicht als unfreundlich empfunden (man sah, daß "etwas passierte"). Die verdeckte Tutor-Eingabe sollte aber trotzdem technisch ermöglicht werden, schon um ihre psychologische Wirkung auf den Nutzer

In Hinsicht auf die vom Tutor aus mögliche Umschaltung der Initiative ist deutlich geworden, daß es nicht ausreicht, dem Nutzer während der Tutoreingabe lediglich die Tasten mit den alphanumerischen Zeichen zu sperren. Hier sollte auf jeden Fall die gesamte Nutzertastatur inklusive Schreibmarkensteuerung und Sendetaste blockiert werden, was dann aber dem Nutzer anzuzeigen ist (z.B. rote Kontrollampe), um Verwirrung zu vermeiden.
Hat man auch ein entsprechendes Signal für die Rückschaltung der Schreibinitiative an den Nutzer, so kann man damit die Schwierigkeit umgehen, die darin besteht, daß er eine erwartete Eingabe oft aus dem Bildschirminhalt (etwa aus der Schreibmarken-position) "herausinterpretieren" muß.

Die Fehlermeldung bei syntaktisch falschen Nutzer(Lerner)-Eingaben schien in dieser Form des LALs durch die Ausgabe einer Leerzeile und das anschließende Warten auf eine neue Eingabe ausreichend zu sein. Dagegen führte die Ausgabe des Textes "WIE BITTE?" häufig zu Verwirrung.
Wie ist das zu erklären?
Da die Nutzer schon nach kurzer Zeit merkten, daß ihre Eingaben (Probleme, Fragen, Verzweigungswünsche, Lösungen von Testaufgaben) in der Regel adaptiv (d.h. auch auf verschieden hohen semantischen Ebenen) beantwortet wurden und besonders im "Lerner-Tutor-Dialog" syntaktische Eingabefehler vom Tutor "übersehen" wurden, konnten sie die (undifferenzierte) Reaktion "WIE BITTE?" (z.B. nach falscher Schlüsselwort-eingabe) nicht hinreichend genau interpretieren.
Sie gingen sogar normalerweise davon aus, daß diese Reaktion sich keinesfalls auf einen syntaktischen Fehler beziehen kann (da sie sich auf ein intelligentes System eingestellt hatten!) und gerieten umso mehr in Verwirrung, je mehr sie Interpretationen über den Sinn (!) dieser Reaktion anstellten.

Als Forderung kann man aus dieser Erfahrung ableiten:

> Die Eingabe-Prüfroutinen müssen intelligenter gemacht werden und/oder die Fehlermeldung muß sich genauer auf die Art des Fehlers beziehen sowie ein detailliertes Ergebnis der Fehleranalyse übermitteln.

5.2.2 Akustischer Dialog

Die Möglichkeit, mit dem Tutor akustisch zu kommunizieren, wurde während der Funkausstellung nicht so häufig in Anspruch genommen wie der alphanumerische Dialog.

Die Gründe hierfür müssen noch untersucht werden. Teilweise mag es daran liegen, daß der Nutzer im Programm selbst keinen Hinweis auf die Möglichkeit eines akustischen Dialogs vorfand.

Aber selbst nach dem Hinweis auf diese Möglichkeit seitens des Personals auf dem HHI-Stand wurde sie selten in Anspruch genommen (ein Besucher: "So kann ich mich ja auch über Telefon unterhalten.").
In Hinsicht auf die zukünftige Beanspruchung des akustischen Kanals sollte man jedoch mit Interpretationen sehr vorsichtig sein.
Hierzu nur zwei Anmerkungen:

1) Die Kommunikation mittels alphanumerischer Tastatur hat für viele potentielle Nutzer des ZKTVs einen typischen Neuigkeits-Effekt, der zur Bevorzugung gegenüber der (bekannten) Kommunikation über einen akustischen Kanal führen kann.
Diese Tendenz wurde auf der Funkausstellung noch dadurch verstärkt, daß die inhaltlichen Probleme des Programms und deren Klärung oft nicht im Vordergrund des Besucherinteresses standen.

2) Das Programm VORFAHRT beinhaltet (zumindest bei einer kurzfristigen Nutzung wie auf einer Ausstellung) nicht so vielschichtige und komplizierte Probleme, daß man die auftretenden Fragen nicht in kurzer und präziser Form über die alphanumerische Tastatur eingeben könnte. Das kann sich bei anderen Programmen ändern, aber auch beim Programm VORFAHRT, wenn der Nutzerzugriff aufgrund eines persönlichen Problems erfolgt oder der Vorbereitung auf die Führerscheinprüfung dient.
Diese Anmerkungen resultieren nicht zuletzt aus den Erfahrungen bei der Durchführung von Live-Dialogen, sie sollten aber in Hypothesenform gebracht und ausgetestet werden.

5.2.3 Breitbandrückkanal

An dieser Stelle sei noch eine kurze Anmerkung zu der Frage erlaubt, ob ein Lehrsystem nicht effektiver und/oder benutzerfreundlicher sei, wenn Lerner und Tutor sich gegenseitig sehen, was bedeutet, daß auch vom Teilnehmer ein Breitbandkanal zur Zentrale geschaltet werden müßte (Bildfernsprechen).

Hierbei wird es u.a. darauf ankommen, allgemeine Aussagen über die Wirkungen menschlicher Ausdruckserscheinungen und spezielle Untersuchungsergebnisse (z.B. zum Problem "sichtbarer Moderator" - Issing/Roth 1969) auf das Problem Zweiweg-Kabelfernsehen zu beziehen und daraus entsprechende Hypothesen abzuleiten.

5.3 Nutzerinitiativen ohne Tutorunterstützung

Die Initiativen der Nutzer in Hinsicht auf Verzweigungen im Programm mittels Schlüsselworteingabe waren selten.
Das lag wahrscheinlich hauptsächlich daran, daß die Verzweigungsmöglichkeiten und die entsprechenden Befehle in den vorhandenen Programmen sehr unterschiedlich waren.
Es war den Besuchern daher in der relativ kurzen Zeit ihres Aufenthalts auf dem HHI-Stand nicht möglich, entsprechende Schemata (vgl. Rumelhart, Ortony 1976) über Teilnehmer-Kommandos zu bilden und anzuwenden.

In der Anregung der Nutzer zur Eigeninitiative auch in Hinsicht auf Programmsteuerung und Informationszugriff liegt aber eine wichtige medienpädagogische Aufgabe, die in einem zukünftigen ZKTV nicht zu kurz kommen darf, wenn dieses Medium eine echte Innovation werden soll.
Dafür muß das HHI-Laborprojekt wichtige Vorarbeiten durch Entwicklung einer für alle Teilnehmer verständlichen und für alle Dienste anwendbaren Kommandosprache leisten.

6 Betrachtungen zur Implementierung

Das Programmsystem "Vorfahrt" wurde in der Höheren Programmiersprache FORTRAN geschrieben. Diese Sprache hat einige gravierende Nachteile: Bei der Programmierung wirkt sich störend aus, daß FORTRAN eine relativ maschinennahe Sprache ist. Der Anwender kann oft nicht so frei programmieren, wie es in Hinsicht auf das zu lösende Problem wünschenswert wäre. Der angebotene Befehlsvorrat und die Befehlsstruktur erlauben dies nicht. Es ist zum Beispiel nicht möglich, dynamische Speicherplatzverwaltung zu erreichen, rekursive und reentrante Programme zu generieren oder Interrupts zu verarbeiten. Insbesondere die letztgenannten Möglichkeiten sind oft notwendig, wenn ein sinnvoll arbeitendes Dialogsystem erstellt werden soll. Bei dem implementierten System fiel insbesondere auch die schlechte Textverarbeitungsmöglichkeit ins Gewicht. Zur Vereinfachung wurde deshalb der Text in das Programm eingearbeitet.

6.1 Software

6.1.1 Textmanipulation

Dieses Einarbeiten des Textes hat natürlich den wesentlichen Nachteil, daß der Text nur umständlich geändert werden kann. Bei Textänderungen muß das Programm geändert, neu compiliert, gebunden und geladen werden. Deshalb wäre es sinnvoll, z.B. mit Textvariablen zu arbeiten, denen dann während der Laufzeit -falls gewünscht- auch ein neuer Text zugewiesen werden kann.

Nun hätte dieses Verfahren allerdings den Nachteil, daß der so geänderte Text nach Beendigung des Programmablaufs verloren wäre. Um dies zu vermeiden, kann man jeden Text auf Platte abspeichern. Dieser Text wird dann beim Laden in den Kernspeicher gebracht. Textänderungen während des Programmlaufs können sowohl in den Kernspeicher geschrieben werden als auch auf Platte abgespeichert werden. Der Programmautor kann dann den alten Text auf der Platte vom neuen überschreiben lassen.

6.1.2 Strukturen erstellen und manipulieren

Mit der oben beschriebenen Methode lassen sich Texte (Daten) gut ändern. Das Programm, d.h. die Struktur (Algorithmus) jedoch ist nicht einfach zu verändern. Nachfolgend soll eine Methode beschrieben

werden, wie ein Programmautor ein Programm mit LHM oder ZAM direkt - also nicht durch einen ausgebildeten Programmierer - erstellen und ändern kann. Diese Änderungen sind auch zur Laufzeit denkbar.

Als erstes hierzu soll zunächst ein "Block" definiert werden:
Ein Block besteht in der Regel aus
- Information (Ausgabe am Endgerät)
- Warten auf Eingabe
- Antwortanalyse

Ein Block ist somit gleichzeitig ein Knoten. Die Blöcke mögen zur übersichtlichen Darstellung eine Numerierung wie in Abb. 2.1 erhalten.

Für einen Block soll der Programmautor folgende Terme definieren (Optionen sind in ( ) gesetzt):

```
1  Block Nr.:
2  Ansprung: (Tt-Name) (,Tn-Name 1, von Block Nr. (+) )
             (,Tn-Name 2, von Block Nr. (+) ) (,Tn-Name 3, von ... ;
3 (Absprung: (TE) (,erlaubter: Tn-Name 1 (,Tn-Name 2 (,...)
                  (,nicht erlaubter: Tn-Name 1 (, Tn-Name 2 (,...);)

4  AV-Definition 1:
4.1 (Bildschirm löschen (von Position) (bis Position);)
4.2 (Standbild Nr.;)
4.3 (Film Nr.;)
4.4 (Ton Nr.;)
4.5 ((Position a,) Text: "a"(,(Position a1,) Text: "a1"(,(... ;)

5  AV-Definition 2:
5.1 (Bildschirm löschen(von Position) (bis Position);)
5.2 (Standbild Nr.;)
5.3 (Film Nr.;)
5.4 (Ton Nr.;)
5.5 ( (Position b,) Text:"b" (,(Position  b1 ) Text: "b1"(,( ...;)

6  AV-Definition 3:
          :

n Ende Block;
```

Tabelle 6 - 1

Zeile 1: Numerierung des Blocks

Zeile 2: Name bzw. Namen des Blocks. Die Namen geben an, unter welchen Namen der Block erreicht werden kann. Dabei ist der Tt-Name eine Zahl (möglicherweise aber auch eine beliebige Zeichenkette) und der Tn-Name eine beliebige Zeichenkette. Der Term "von Block Nr." gibt an, welchem Block dieser Tn-Name bekannt sein soll. Die Option + bedeutet, daß der Tn-Name ab dem benannten Block auch jedem hierarchisch tiefer liegenden Block bekannt sein soll. Der Tt-Name dagegen soll jedem Block bekannt sein.

Beispiele hierzu:

```
Block 0.2.1;
Anspr: "    ", 0.2 ;
        :
Block 0.1.1.2;
Anspr: 102, "NEIN", 0.1.1 ;
        :

Block 0.1.1.1.1 ;
Anspr: 100, "AB", 0.1+,"AA",0+,"AB",0+;
        :
```

Struktur hierzu:

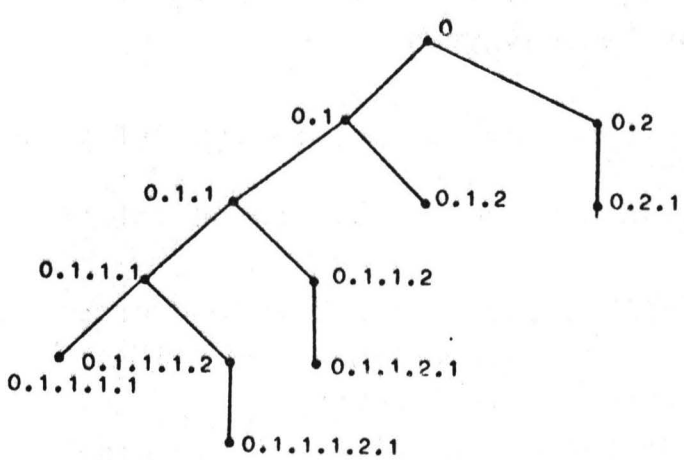


Abb. 6.1 Strukturbeispiel

Block 0.2.1 kann nur von Block 0.2 durch Absenden einer Leerzeile erreicht werden.

Block 0.1.1.2 kann durch die Tt-Eingabe "102" von jedem Block aus oder durch die Tn-Eingabe "NEIN" von Block 0.1.1 erreicht werden.

Block 0.1.1.1.1 kann von allen Blöcken unter dem Tt-Namen "100" erreicht werden. Von den Blöcken des Astes 0.1.1 kann der Block unter dem Tn-Namen "AB", von denen des Astes 0.1 unter dem Namen "AA" und denen des Astes 0.2 wiederum unter dem Namen "AB" erreicht werden. Da Ast 0.1.1 ein Teil des Astes 0.1. ist, kann z.B. Block 0.1.1.1.1 von 0.1.1.1.2.1 entweder durch den Tn-Namen "AA" oder durch "AB" erreicht werden.

Denkbar ist weiterhin, daß eine ausführlichere Textanalyse vorgenommen werden soll (freier Text). In diesem Falle könnte in dieser Zeile eine entsprechende Spezifikation vorgenommen werden.

Zeile 3: Falls diese Zeile nicht definiert wird, kann der Block mit jedem ihm bekannten Tn-Namen (Definitionen in den Zeilen 2 der anderen Blöcke) verlassen werden. Durch die Angabe nicht erlaubter Namen wird die Menge der Namen eingeschränkt. Der Block kann mit diesen angegebenen Namen nicht verlassen werden. Werden dagegen erlaubte Namen angegeben, dann kann der Block nur mit diesen Namen verlassen werden. Die Option TE (Tutoreneingabe) bedeutet, daß für häufig wiederkehrende Einschränkungen wie beim LTD auch vorher definierte Abkürzungen verwendet werden können. Tt-Namen sollen stets in der gesamten Struktur bekannt sein.

Zeile 4: AV-Beiträge und Texte (Präsentationen) werden definiert:

4.1 Text auf dem Bildschirm oder Teile davon sollen gelöscht werden
4.2 Angabe des Standbildes, das eingeblendet werden soll
4.3 Angabe des Filmes, der eingeblendet werden soll
4.4 Angabe der akustischen Einblendung
4.5 Definition des Textes mit möglichen Angaben der Anfangsposition des Textes, der ausgegeben werden soll. Im Programmablauf wartet das Programm nach Ausgabe des

4.5 Textes dieser Zeile auf eine Eingabe. An dieser Stelle ist ein Absprung, jedoch kein Ansprung möglich.

Zeile 5: wie 4. Es können beliebig viele Präsentationen definiert werden.

n: Beenden des Blockes

Ein Bild kann somit wie im folgenden Beispiel definiert sein:

```
1   Block 0.3.2.4.1;
2   Anspr.: "   ", 0.3.2.4;
3   Abspr.: TE, erl: "JA", "NEIN", nichterl: "ENDE";
4   Bildschirm löschen von 10,1 bis 20,20;
5   Dia Nr. 27;
6   Position 5,5;
7   Text: "Möchten Sie ein Beispiel sehen?",
8   Position 8,7, Text: "JA",
9   Position 10,7, Text: "NEIN";
10  Ende Block;
```

Kommentar hierzu:

Zeile 2: Der Block ist nur vom Block 0.3.2.4 durch Absenden einer Leerzeile zu erreichen.
Zeile 3: Der Tn-kann mit TE-Befehlen (dazu gehöre u.a. der Befehl "ENDE") jedoch nicht mit "ENDE" aber zusätzlich mit "JA" und "NEIN" den Block verlassen.
Zeile 4: Der Bildschirm soll im angegebenen Bereich gelöscht werden
Zeile 5: Das Dia Nr. 27 soll eingeblendet werden.
Zeile 6-9: Positions- und Textangaben.

Mit diesen Blöcken lassen sich sehr komplexe Strukturen erstellen. Diese Strukturen lassen sich leicht verändern, da der Inhalt eines Blockes leicht geändert werden kann. Auch lassen sich Blöcke ohne weiteres entfernen oder einfügen. Ebenso ist es möglich, daß zunächst nur die Struktur erstellt wird, diese ausgetestet wird und AV-Beiträge und Texte erst danach eingefügt werden. Als Erweiterung ist denkbar, daß die Struktur auch von den Vorkenntnissen des Tns abhängig ist, daß also bestimmte Blöcke "umsprungen" oder daß andere Blöcke angesprungen werden. Zur Steigerung der Flexibilität sollte eine relativ freie Programmierung zusätzlich möglich sein (vgl. Inline - Assembler).
Dem Programmautor soll ein System, das diese Blöcke verarbeiten kann, umfangreiche Hilfen zur Verfügung stellen. Es sollte z.B. nicht an eine Numerierung der Blöcke gebunden sein, wie sie hier benutzt wird, sondern er sollte beliebige Zeichenketten dafür angeben können. Weitere Hilfen wären z.B.:

- Darstellen und Zeichnen der Struktur (mit Hilfe des Autors oder vom Programm)
- Hilfen beim Erstellen (Editieren) der Blöcke
- Hilfen beim Austesten ...

Mit diesen Blöcken kann also ein Programmautor ohne Programmierkenntnisse komplexe Programme mit LHM oder ZAM erstellen, die Text, Dias und andere AV-Beiträge darbieten und eine Kommunikation zwischen Tn und Rechner und Tn und Tt ermöglichen. Änderungen des Textes und der Struktur sind leicht möglich.

6.1.3 Programmierung und Sprache

Dem Programmautor steht also für die oben genannten Zwecke eine Autorensprache (AS) zur Verfügung. Diese AS sollte in eine andere existierende höhere Programmiersprache umgesetzt werden. Dies erscheint nicht sehr schwierig (s.CDL). Die Sprache kann nun z.B. vereinfacht werden, wenn man keine Namen zuläßt, die nur in bestimmten Ästen bekannt sind, oder wenn man höchstens einen Tn-Namen für den Block zuläßt und dieser dann in der ganzen Struktur bekannt sein soll. Im letzteren Fall müssen Verzweigungen wie "JA" und "NEIN" jedoch in den Blöcken angegeben werden, von denen aus verzweigt wird.

Es gibt verschiedene Möglichkeiten, Sprünge zu organisieren. Die eine wäre die wie im Programm "Verkehr" über Verteiler. Eine andere, elegantere und schneller ausführbare ist die, jedem Namen die zugehörige Kernspeicheradresse zuzuordnen. In diesem Fall erfolgt der Sprung direkt in den entsprechenden Programmteil und nicht über den Umweg über Verteiler.
Beispielsweise kann ein Programmausschnitt hiezu folgendermaßen aussehen:

```
1 NAME 1 : NAME 2: CALL INHALT (....)
2                  CALL ANALYSE (...,Label,...)
3                  GO TO Label
```

Zeile 1 dieses Beispiels kann unter den Namen (Label) NAME 1 und NAME 2 angesprungen werden. Es wird UP INHALT aufgerufen, das Texte und andere AV-Beiträge darstellt. Das UP ANALYSE analysiert die nachfolgende Eingabe und organisiert den entsprechenden weiteren Programmablauf. Bedeutet die Eingabe, daß ein weiterer Block abgerufen werden soll, wird "Label" der entsprechende, eingegebene Name zugewiesen und in Zeile 3 der Sprung ausgeführt.

Dieses Schema kann beliebig erweitert werden. Jeder Block kann dann an irgendeiner Stelle zwischen anderen Blöcken stehen. D.h., aus dem Programm selber ist die Struktur nicht mehr zu erkennen.

Eine andere Möglichkeit wäre, die Namen des Unterprogramms, das die nächsten Präsentationen darstellen soll, aus dem Block, den das Programm gerade bearbeitet, und aus den Tn- oder Tt-Befehlen zu bestimmen.
Beispiel:

```
      1   CALL BESTIMMUP (..., UPINHALT,...)
          CALL UPINHALT (...)
          GO TO 1
```

Das heißt, das Hauptprogramm besteht im wesentlichen aus diesen 2 CALL-Befehlen.

Geht man von der Voraussetzung aus, daß der Plattenverkehr und die Mehrfachausnutzung des Programms vom Betriebssystem optimal organisiert wird, dann hat die zu wählende Höhere Programmiersprache u.a. folgende wichtige,nicht von allen Programmiersprachen erfüllte

Bedingungen zu erfüllen.

- einfache Textverarbeitung
- einfache Erstellung aus einer AS

Vorhandene Höhere Programmiersprachen müssen somit auf ihre o.a. Eignung und auf ihre Verfügbarkeit (am Rechner) untersucht werden.

6.1.4 Dienste (2)

Nun soll allerdings nicht nur ein Tn mit dem Programmsystem arbeiten können, sondern es soll möglich sein, daß mehrere Tn gleichzeitig mit diesen oder auch anderen Systemen arbeiten. Diese Möglichkeit ist dann gegeben, wenn das Betriebssystem des verwendeten Rechners Time-Sharing-Betrieb zuläßt. Da es möglich sein soll, daß eine große Anzahl von Tn mit dem Rechner korrespondieren , muß das Dienste-Lauf-System (DILS) entsprechend konzipiert sein.
Für ein solches System könnte folgendes Konzept (betrachtet für ein Programmsystem) vorgesehen werden:

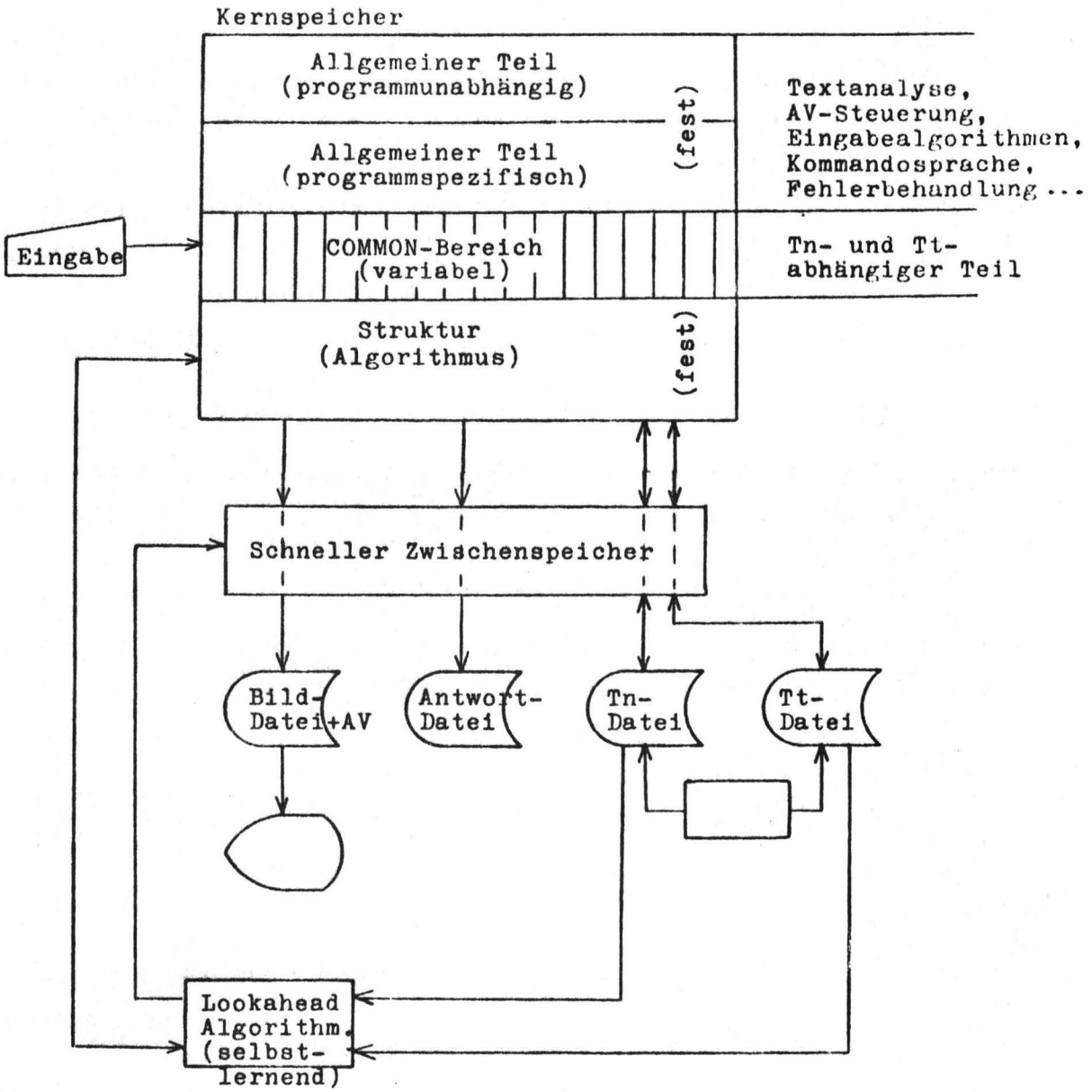


Abb. 6.2 Konzept für ein DILS

Der Kernspeicher sei in drei Bereiche aufgeteilt.

Bereich 1: Hier befinde sich ein allgemeiner Teil, der unabhängig von dem Programmsystem ist, und einer, der programmspezifisch, jedoch unabhängig von der Struktur ist. Hierzu gehören z.B.: Textanalyse, AV-Steuerung, Eingabealgorithmen, Kommandosprache, Fehlerbehandlung, u.a.m., die jeweils zu einem der beiden allgemeinen Teile oder zu beiden gehören können.

Bereich 2: Ein Common-Bereich, der jedem Tn zugeordnet ist und in dem Tn- und Tt- spezifische Angaben stehen. Diese können sein: Zustand, in dem sich das Programm befindet, Speicheradressen der IO-Buffer von Tn und Tt, Tn-spezifische Strukturdefinitionen, u.a.m. In diesem Bereich stehen also Daten, die von Tn (und auch Tt) und vom Programmablauf abhängig sind und damit variabel sind.

Bereich 3: Hier befinde sich die eigentliche Struktur (Algorithmus) des Programms (Knoten, Blöcke ohne AV-Inhalt)

In den Bereichen 1 und 3 stehen nur feste Daten.

Neben dem Kernspeicher gibt es noch einen schnellen Zwischenspeicher. Hier werden Daten (AC-Inhalte, Text) bereitgestellt, die gerade benötigt werden und Daten, die wahrscheinlich als nächstes benötigt werden. Die Bereitstellung dieser Daten geschieht durch einen selbstlernenden Lookahead Algorithmus. Welche Daten bereitgestellt werden, ergibt sich aus dem Zustand des Programms (Struktur) und dem Gelernten. Die zum Lernen notwendigen Informationen werden aus der Tn-Datei und aus der Tt-Datei geholt. In diesen Dateien stehen aufbereitete Verhaltensdaten (semantisches Netz, Statistik) von Tn bzw. Tt. Befindet sich also der Tn an einem bestimmten Knoten, dann "sieht" dieser Algorithmus noch, welchen Knoten vorhergehende Tn (oder auch der Tt) nach diesem Knoten am häufigsten angesprungen haben, und sorgt dann dafür, daß die entsprechenden Daten von der Platte in den schnellen Zwischenspeicher gebracht werden. Somit ist in den meisten Fällen eine kurze Antwortzeit gewährleistet.

Antwort-Datei und Tn- und Tt-Datei werden zur Auswertung benötigt. Es können also sowohl Tn- und Tt-Verhalten als auch deren Antworten (Text) analysiert werden.

Tn- und Tt-Datei müssen wahrscheinlich in irgendeiner Weise miteinander verknüpft sein, damit zum einen der Lookahead-Algorithmus für ihn relevante Daten aus den Dateien holen kann und zum anderen, damit das Tn- und Tt-Verhalten im Vergleich mit Verhaltensweisen früherer Tn's und Tt's in den Programmablauf mit einwirken kann.

Außer einem DILS muß noch ein Dienste-Entwicklungs-System (DES) existieren, das Hilfen zum Erstellen (wie z.B. AS) und Prüfen von Programmen bereitstellt. (2)

Das Betriebssystem (BS), das diese Dienste leistet, kann kein allgemein verwendungsfähiges BS sein wie z.B. BS 2000, da die speziellen Anforderungen von einem solchen BS in vielen Fällen nicht oder nur zeitaufwendig und umständlich zu erfüllen sind.

---

(2) siehe auch: K. Haefner, Bemerkungen zur KTV-Software-Organisation, Juni 1977

6.2 Courseware

Ein wichtiger Dienst im Bildungsbereich des ZKTV-Systems könnte die Erstellung und Evaluierung von Courseware werden.
Abschließend deshalb noch einige Bemerkungen zur Courseware-Forschung und -Entwicklung aus der Sicht des Mediendidaktikers.

6.2.1 Forschung

Daß Lehrprogramme auch in einem ZKTV-System inhaltlich hinreichend evaluiert werden müssen, ist eigentlich eine Selbstverständlichkeit. - Im Zusammenhang mit der konkreten Implementierung in einem solchen System existieren aber noch einige spezielle Fragen hinsichtlich der besonderen Interaktionsstrukturen im Kabelfernsehen, die ebenfalls im Evaluationskonzept berücksichtigt werden sollten.
Ausgehend von den unzureichenden, schwer zu verallgemeinernden Erfahrungen auf der Internationalen Funkausstellung (vgl. Abbschnitt 5) sollte beispielsweise untersucht werden, wie häufig der menschliche Berater tatsächlich vom Lernenden verlangt wird und - fast noch wichtiger - welche Forderungen an ihn herangetragen werden. Bei der Untersuchung dieser - eventuell auch für andere Dialogdienste - wichtigen Frage kann man von einer "Lerner-" und von einer "Tutorposition" herangehen.
In Hinsicht auf die "Lernerposition" wären Fragen zu klären wie:

- Welche Art von Fragestellungen bevorzugen die Lernenden (Frageformulierung) bzw. gibt es einen Zusammenhang zwischen der Frageformulierung und dem Inhalt und/oder der Struktur des Programms?

- welche speziellen Anforderungen stellen Lernende (z.B. Anforderung von Präsentationsmaterial, Beispielen, Übungen usw.), welche Zusammenhänge kann man hier eruieren? Wenn es solche Zusammenhänge gibt, wie kann man sie im ZKTV berücksichtigen, um das System benutzerfreundlicher zu machen?
- wie häufig und wozu wird der akustische Kanal in Anspruch genommen, gibt es hier Affinitäten zu bestimmten Inhalten oder Strukturen?

usw.

Die mit der Tätigkeit des Tutors zusammenhängenden Fragen gehen etwa in folgende Richtung:

- wie differenziert (mit welchem Auflösungsgrad) müssen Informationen bei welchen Lernenden dargeboten werden?
- wie differenziert müssen Lösungen von Testaufgaben evaluiert werden? Welche Lösungshilfen müssen geboten werden? Inwieweit können entsprechende Funktionen in einem zukünftigen ZKTV-System vom Rechner übernommen werden?
- Nach welchen Kriterien setzt ein Tutor Prioritäten, sowohl in Hinsicht auf verschiedene Fragen eines Lerners als auch in Hinsicht auf Fragen verschiedener Lerner ("Warteschlangentheorie" für den Tutor)?

Die Untersuchung des letzten Punktes sollte möglichst auch schon vor der Installation mehrerer Terminals versucht werden (evtl. durch Simulation im Live-Dialog), um Kriterien für eine sinnvolle Hard- und Softwarekonzeption in diesem Bereich zu gewinnen.

Es wird also darauf ankommen, Tutor-Strategien, wie sie speziell für das ZKTV entwickelt werden müssen, zu untersuchen, um dann Entscheidungen hinsichtlich der Objektivierung von Tutoroperationen treffen zu können (z.B. Auswertroutinen der Lerneingaben zur Gedächtnisstütze des Tutors).

Bei der Untersuchung von Tutor-Strategien sollte sowohl die Effektivität als auch die Akzeptanz (Benutzerfreundlichkeit) eine Rolle spielen.

Ein möglicher Unteraspekt wäre beispielsweise die Frage, ob sich Lernende (oder allgemein Nutzer) unterschiedlich verhalten, je nachdem ob sie wissen, daß ein Mensch bestimmte Interaktionen steuert oder ob sie der Meinung sind, nur mit einem Rechner zu kommunizieren (Bobrow et al. 1976).
Hierzu könnte eine Art "Placebo-Versuch" stattfinden, bei dem Versuchspersonen einmal mit und einmal ohne ihr Wissen mit einem menschlichen Berater über das Medium ZKTV kommunizieren.

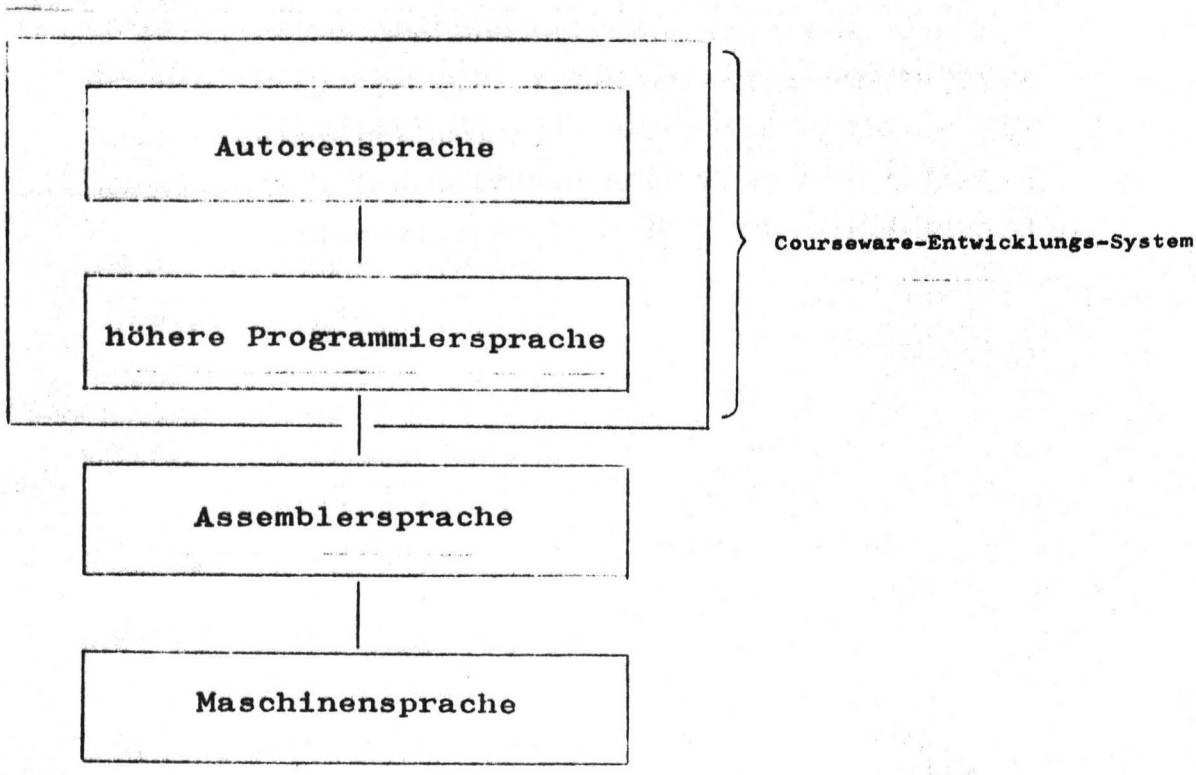
Bei allen hier angesprochenen Forschungsfragen sollte überprüft werden, inwieweit sich Forschungsmethoden und -ergebnisse verwandter Disziplinen (z.B. "Learner Control Language" - An Overwiew of the TICCIT Program, 1976, "ATI-Forschung" - Schwarzer, Steinhagen, 1975, usw.) sinnvoll integrieren lassen.

6.2.2 Entwicklung

Die Courseware-Entwicklung kann unterschiedlich organisiert werden; ein größeres Programmsystem wird in der Regel von einem Autorenteam erstellt. Es sollte aber beachtet werden, daß ein einfach zu handhabendes Courseware-Entwicklungs-System auch bei engagierten Lehrern auf Interesse stoßen wird. Diese würden sich dann eventuell kleine Programme bzw. Programmsegmente erstellen, die sie gezielt im Sinne ihrer didaktischen Intentionen benutzen wollen. Die Arbeitsüberlastung der Lehrer kann nur dann als Gegenargument gegen eine solche Perspektive gelten, wenn eine Courseware-Entwicklung im ZKTV-System nicht auch mit einem Minimum an Aufwand und Vorkenntnissen (auch im Verhältnis zu anderen Unterrichtsmedien) möglich ist.

Bei der Implementierung von Lehrprogrammen sollte die Offenheit des Systems hinsichtlich unterschiedlicher Lehrstrategien beibehalten werden. Gerade in einer Phase, in der nach der Kritik an der behavioristischen Lerntheorie und an der auf ihr aufbauenden Instruktionsstrategie erst in Ansätzen Modelle für Lehrstrategien auf der Grundlage kognitiver Theorien entwickelt werden (z.B. Collins et al. 1975), könnte sich eine Festschreibung bestimmter didaktischer Modelle durch entsprechende technische Restriktionen auf die langfristige Perpektive des ZKTVs nachteilig auswirken.
Es soll dabei keinesfalls übersehen werden, daß für die Entwicklung in einem Laborsystem bestimmte Entscheidungen (auch unter finanziellem Aspekt) getroffen werden müssen.

Es geht also darum, einen Kompromiß zu finden, der einerseits finanziell vertretbar ist und andererseits eine optimale Flexibilität des Systems gewährleistet.
Die geforderte Flexibilität im Bereich der Bildungs- oder Lerndienste kann dadurch gewährleistet werden, daß man Lehrprogrammautoren verschiedener didaktischer "Schulen" (evtl. auch "didaktische Laien") dazu anregt, Lehrprogramme für ein ZKTV zu erstellen.
Da viele dieser Autoren keine Programmiersprache beherrschen (und gerade Courseware dieser Autoren könnte sehr interessant sein), wäre es sinnvoll, ein Courseware-Entwicklungs-System zu entwickeln, das die Programmerstellung auch den Autoren ohne Informatik-Kenntnisse ermöglicht.
In diesem System sollten aber auch Autoren mit Programmiererfahrung höhere Programmiersprachen anwenden können.
Ein solches Courseware-Entwicklungs-System würde daher eine Integration von Autorensprache und höherer Programmiersprache leisten müssen.
Inwieweit sich daraus ein allgemeines Dienste-Entwicklungs-System konstruieren läßt, bleibt weiteren Diskussionen vorbehalten.



Wenn "Nicht-Programmierer" Courseware erstellen, so werden sie zunächst (mehr oder weniger explizit) von bestimmten didaktischen Modellen ausgehen. Am Anfang der Arbeit steht dabei meist die Lehrstoff-strukturierung und -detaillierung (z.B. Frank/Meder 1971, König/Riedel 1975, Klauer 1974, Heimann u.a. 1970).

An dieser Stelle sollte den Autoren größtmögliche Freiheit gelassen werden.
Aus den Erfahrungen mit dem Programm VORFAHRT wäre höchstens die Empfehlung der Zuordnung von "importance-tags" (Collins et al. 1975) zu bestimmten Informationseinheiten zu geben.

Der Autor sollte sich also u.a. folgende Überlegungen machen:

- welche Informationen können vorausgesetzt werden?
- welche Informationen sind eventuell bekannt?
- welche Informationen sind wahrscheinlich nicht bekannt,aber wichtig?
- welche Informationen sind wahrscheinlich nicht bekannt, aber unwichtig?

Hieraus ergeben sich dann entsprechende Prioritäten bei der Zuordnung von Präsentationsmaterial in verschiedenen Phasen des Lernprozesses.
Die Kenntnis der entsprechenden Prioritäten ist u.a. für eine adäquate Reaktion auf Lerner-Kommandos wie LEICHTER und SCHWIERIGER notwendig.

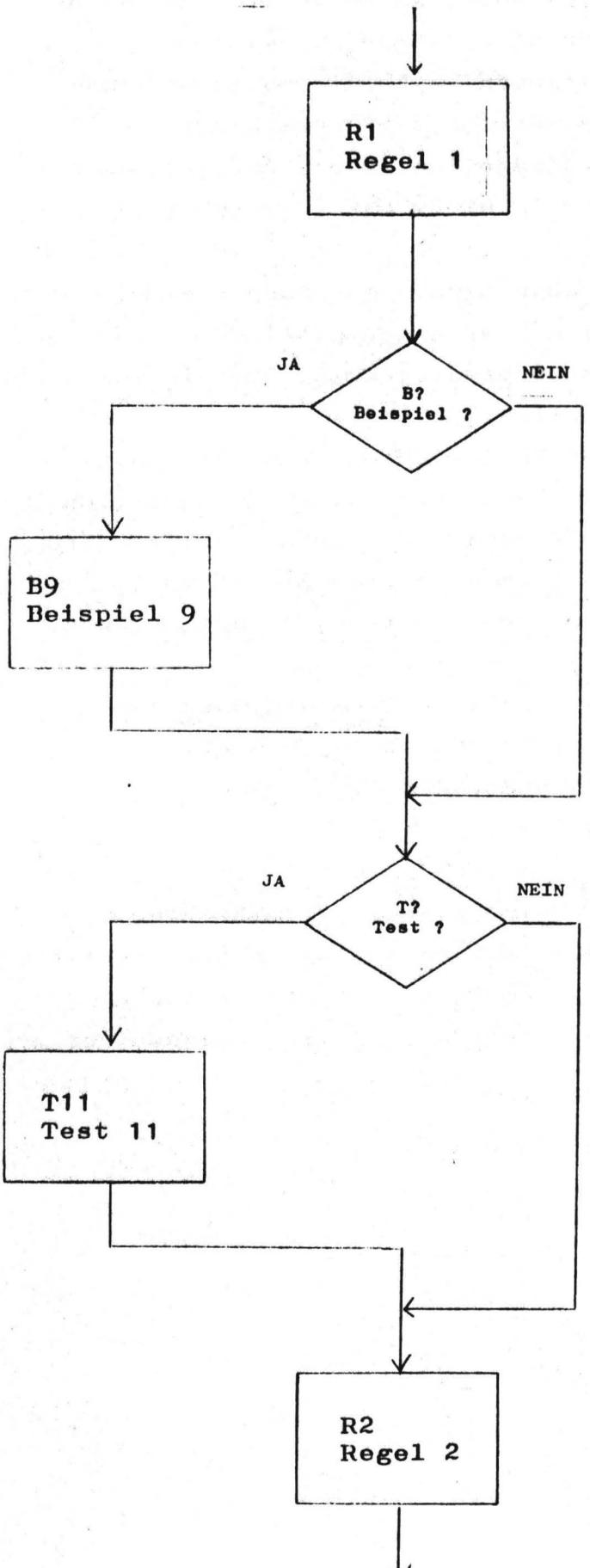
Nach der Strukturierung des Inhalts sollte der Autor didaktische Voraussetzungsbeziehungen zwischen verschiedenen Informationen bzw. Informationsblöcken untersuchen und diese möglichst in Form eines Voraussetzungsnetzes darstellen. Diese Darstellung ist auch für den Tutor in einem LAL sehr wichtig.

Ausgehend von den Voraussetzungsbeziehungen macht der Autor sich Gedanken über einen möglichen Programmablauf und versucht diesen in Form eines Flußdiagramms darzustellen.
Geht der Autor von der Philosophie eines LALs aus, dann kann dieses Fluß- bzw. Ablaufdiagramm natürlich nicht alle möglichen Verzweigungen enthalten, sondern nur die die sich aus einem programmgesteuerten (im Gegensatz zum lernergesteuerten-) Ablauf ergeben.

Den einzelnen Informationsblöcken werden dann vom Autor frei wählbare Zeichenketten zugeordnet, bei denen auch mnemotechnische Gesichtspunkte eine Rolle spielen sollten.

Um die vorgeschlagene Methode kurz zu erläutern, wird im folgenden bewußt ein sehr einfach strukturiertes Ablaufdiagramm vorausgesetzt, wie es auch Autoren erstellen können, die keinerlei Informatikkenntnisse besitzen.

Abb. : Auszug aus einem hypothetischen Ablaufdiagramm



Im obigen Fall will also ein Autor eine Regel vermitteln, dann den Lerner fragen, ob dieser ein Beispiel und/oder eine Testaufgabe zu dieser Regel sehen will, um danach zur nächsten Regel überzugehen.

Nach der Erstellung des Ablaufdiagramms muß der Autor die einzelnen Informationsblöcke spezifizieren. Diese Spezifikation enthält außer der entsprechenden Präsentation auch Aussagen über den Zugriff zu diesem Block sowie über mögliche Einschränkungen beim Verlassen (s. auch 6.1.2).

Die Spezifizierung der einzelnen Informationsblöcke kann man sich bis hin zu bestimmten Formblättern oder Courseware-Entwicklungs-Programmen vorstellen. An dieser Stelle soll nur auf einige funktionale Komponenten Bezug genommen werden.
Funktional beinhaltet die Spezifizierung der Informationsblöcke wie gesagt zwei Aspekte:
1) Präsentationsspezifizierung
2) Strukturspezifizierung

Punkt 1 beinhaltet die Spezifizierung des Präsentationsmaterials, Punkt 2 Aussagen über die Stellung des Blockes in der Programmstruktur.
Die Aussagen über die Programmstruktur kann man sich einerseits in Form eines "Kausalnexus", andererseits in Form eines "Finalnexus" (König/Riedel 1975, S. 125ff) vorstellen.
"Kausalnexus" würde hier bedeuten, daß der Autor für jeden Informationsblock alle möglichen folgenden Blöcke inklusive der entsprechenden Ansprungbefehle spezifizieren müßte (IF-Abfragen).
Da ein LAL über eine große Variationsweite von Verzweigungsmöglichkeiten verfügt, scheint dieses Verfahren dafür sehr umständlich zu sein.

Es sollte daher überlegt werden, ob man die Struktur des Programms nicht in einer Art "Finalnexus" spezifizieren kann.
"Finalnexus" würde hier bedeuten, daß die Ansprung-befehle für die einzelnen Blöcke aufgelistet werden und nur in Ausnahmefällen das "Verlassen" des Blockes spezifiziert wird.
Ein Programmautor hätte also für jeden Block festzulegen:
- Präsentationsmaterial
- Ansprungbefehle (Absprungmöglichkeiten nur in Ausnahmefällen)

Die hier vorgeschlagene Methode soll an einigen (hypothetischen) Beispielen veranschaulicht werden (vgl. dazu das Ablaufdiagramm).
In einem Courseware-Entwicklungs-System müßten u.a. die Präsentationsmöglichkeiten noch weiter spezifiziert werden.

Beispiele:

Block: B?

1. Präsentation
1.1. Rechnerschrift u. Graphik: /
1.2. Standbild: Dia 38 (mit Text: "Möchten Sie zu Regel 1 ein Beispiel sehen?"
1.3. Bewegtbild: /
1.4. Ton: /

2. Ansprung
2.1. Teilnehmer-Befehle: /
2.2. Tutor-Befehle: /
2.3. Befehle von anderen Blöcken: Leerzeile (LZ) von R1

3. Absprung: /

Diese sehr einfache Spezifizierung würde beispielsweise bedeuten:

- Ein Teilnehmer kommt nach der Präsentation von Regel 1 durch die Eingabe einer Leerzeile (und nur so!) zum Block "B?", in dem ihm durch Präsentation eines entsprechenden Standbildes die Frage gestellt wird: "Möchten Sie zu Regel 1 ein Beispiel sehen?" (vgl. Ablaufdiagramm).
- Über den Absprung wird keine Einschränkung definiert.

Hierin zeigt sich ein Vorteil der "finalen" Spezifizierung: Würde man alle von diesem Block aus möglichen Verzweigungen definieren, so müßte man sowohl die Befehle "JA" und "NEIN" als auch alle anderen an dieser Stelle erlaubten Verzweigungsbefehle auflisten.

Block: B9

1. Präsentation
1.1. Rechnerschrift u. Graphik: Formblatt 78 (mit der Beispielerläuterung)
1.2. Standbild: Dia 45 (Foto einer Realsituation)
1.3. Bewegtbild: /
1.4. Ton: Cassette 3, 0000-0030 (Begleitmusik, Geräusche)

2. Ansprung
2.1. Teilnehmer-Befehle: BEISPIEL ZU REGEL 1
2.2. Tutor-Befehle: B9
2.3. Befehle von anderen Blöcken: JA von B?

3. Absprung:

Hier würde spezifiziert, wie auf Block "B9" zugegriffen werden kann (Beispiel durch Präsentation von Bild, Text und Ton):

- Der Teilnehmer kommt von Block "B?" (vgl. Ablaufdiagramm) durch die Eingabe von JA zur Präsentation des Beispiels.
- Der Teilnehmer kommt von jeder Stelle des Programms (außer bei Einschränkung des Absprungs) aus durch die Eingabe von "BEISPIEL ZU REGEL 1" zu dieser Präsentation.
- Der Tutor kann dem Teilnehmer dieses Beispiel an jeder Stelle des Programms durch die Eingabe von "B9" präsentieren.

Block: T11

1. Präsentation
1.1. Rechnerschrift u. Graphik: /
1.2. Standbild: Dia 3 (Testsituation und Testfrage)
1.3. Bewegtbild: /
1.4. Ton: /

2. Ansprung
2.1. Teilnehmer-Befehle: /
2.2. Tutor-Befehle: /
2.3. Befehle von anderen Blöcken: JA von T?

3.Absprung: nur Tutor-Befehl

In diesem Falle würde der Autor die Präsentation der Testaufgabe (Dia) nur durch die Teilnehmer-Eingabe von "JA" im Block "T?" zulassen. Natürlich könnte in einem LAL auch der Tutor im Block "T?" die Eingabe "JA" machen, dann aber nicht verdeckt.

Der Absprung aus dem Block T11 wäre hier nur durch eine Tutor-Eingabe möglich. Das kann beispielsweise dann sinnvoll sein, wenn die Bewertung der Lerner-Antwort auf eine Testaufgabe sehr kompliziert und daher vom Rechner nicht (oder noch nicht) zu leisten ist.
Der Autor sollte in diesem Falle die Kriterien für Entscheidungen über die Richtigkeit der Lerner-Antwort mit angeben.

Die vorgestellten Beispiele sind bewußt sehr einfach gewählt. Hier sollte lediglich das Prinzip veranschaulicht werden.
Die weiteren Spezifizierungen eines solchen Systems bleiben weiteren Arbeiten vorbehalten.
Anzustreben wäre ein Courseware-Entwicklungs-System, in dem der Autor während der Programmerstellung zwischen Autor-Modus und Lerner-Modus wechseln kann.
Im Sinne einer intensiven Entwicklungsevaluierung von Lernprogrammen, sollte man dem Lerner das Einschreiben von Kommentaren in einen elektronischen Briefkasten ermöglichen.

Bleibt noch zu erwähnen, daß für das ZKTV eine angemessene Teilnehmer-Kommandosprache zu entwickeln ist, die bestehen sollte aus:

- festen, für alle Dienste gültigen Kommandos (evtl. entsprechende Kommando-Tasten).
- von Autoren für spezielle Dienste oder Programme definierten Kommandos
- von Teilnehmern zu definierenden Kommandos.

7 Literatur

Atkinson, R.C.: Formal Models of the Learning Process
GPI-Kongress "Lehrsysteme 72",
Berlin 1972

Bobrow, D.G.; Kaplan, R.M.; Kay, M.; Norman, D.A.;
Thompson, H.; Winograd, T.: GUS, A Frame Driven Dialog System
o.O., May 21, 1976

Boeckmann, K.; Lehnert, U.: Bildungstechnologie -
Schwerpunkte der Forschung
und der Diskussion
In: Fortschr. u. Erg. d. Bildungst. 3,
1975, S. 13-17

Bunderson, C.V.: The TICCIT Learner Control Language
Occasional Paper No. 5, Utah 1975

Collins, A.; Warnock, E.H.; Passafiume, J.J.:
Analysis and Synthesis of Tutorial
Dialogues
In: G.H. Bower (Ed.): The Psychology
of Learning and Motivation (Vol. 9),
New York 1975, S. 49-87

Döring, K.W.: Lehr- und Lernmittel: Medien
des Unterrichts
Weinheim und Basel 1973
(2., überarbeitete und erweiterte
Auflage)

Dohmen, G.: Die Realisierung neuer Organisations-
formen des "lebenslangen" Lernens
mit Hilfe moderner Medien
In: Arlt/Issing (Hrsg.): Ergebnisse
und Probleme der Bildungstechnologie
Berlin 1976, S. 21-27

Eyferth, K.: Psychologische Aspekte des CUU
In: Fortschr. u. Erg. d. Bildungst. 3,
1975, S. 107-118

Eyferth, K.; Fischer, K.; Kling, U.; Korte, W.;
Laubsch, J.; Löthe, H.; Schmidt, R.; Schulte, H.;
Werkhofer, K.: Computer im Unterricht
Stuttgart 1974

Frank, H.G.; Meder, B.S.: Einführung in die kybernetische
Pädagogik
München 1971

Haefner, K.: Aspekte der Nutzung des Kabelfernsehens
im Bildungswesen
Unveröffentl. Manuskript, HHI
Berlin, März 1977

Heimann, P.; Otto, G.; Schulz, W.: Unterricht - Analyse und Planung
Hannover 1970 (5. bearbeitete Auflage)

Hilgard, E.R.; Bower, G.H.: Theories of learning
Englewood Cliffs, N.J., 1975

Issing, L.J.: Evaluierung von Unterrichtsmedien
In: Issing/Knigge-Illner (Hrsg.): Unterrichtstechnologie und Mediendidaktik
Weinheim und Basel 1976, S. 141-150

Issing, L.J.; Roth, H.: Ein- und zweikanalige Informationsdarbietung im Schulfernsehen
In: programmiertes Lernen und programmierter Unterricht, Sonderdruck aus 2/1969, S. 76-82

Issing, L.J.; Hannemann, J.; Mühlbach, L.; Gekeler, M.: Untersuchung von Lehr- und Lernprozessen in einem Lerner-adaptiven Lehrsystem
HHI, Berlin 1977

Judd, W.A.; Bunderson, C.V.; Bessent, E.W.: An investigation of the effects of learner control in computer-assisted instruction prerequisite mathematics (maths)
Technical Report No. 5, The University of Texas, Austin, Texas 1970

Klauer, K.J.: Methodik der Lehrzieldefinition und Lehrstoffanalyse
Düsseldorf 1974

König, E.; Riedel, H.: Unterrichtsplanung I
Weinheim und Basel 1975

Leontjew, A.N.: Probleme der Entwicklung des Psychischen
Frankfurt am Main 1973

Miller, G.A.; Galanter, E.; Pribram, K.H.: Strategien des Handelns. Pläne und Strukturen des Verhaltens
Stuttgart 1974

An Overview of the TICCIT Program
The MITRE-Corporation, July 1976

Rumelhart, D.E.; Ortony, A.: The Representation of Knowledge in Memory
Technical Report No. 55,
University of California,
San Diego, La Jolla, January 1976

Schwarzer, R.; Steinhagen, K. (Hrsg.): Adaptiver Unterricht
München 1975