A SYSTEM TO PRODUCE CAI COURSES
FROM LOGICALLY STRUCTURED
EDUCATIONAL MATERIAL

by

Luis Osin

Technical Report No. 29
November 1973
A B S T R A C T

A system for automatic production of Computer-Aided Instruction (CAI) courses is described. Input to the system consists of a collection of documents plus structured information supplied by the author. The derived sequential organization is based on a similarity distance between documents. Presentation operates in an adaptive mixed-initiative mode. Current results show that course programming time is reduced by one to two orders of magnitude compared to conventional CAI methods.
1. INTRODUCTION

In spite of the advantages that CAI offers, its use is so restricted that up to now it is generally developed in a "laboratory environment". It is enough to see that the number of universities that provide CAI courses is insignificant compared to the number of those that do not offer them.

Is it because of cost? Not really. The prospective cost is less than that required for classical teaching [1]. Are the results too poor? On the contrary, we already get reports of better results obtained with CAI than with conventional procedures [2], and students feel better and prefer working with self-paced methods [3].

Then, what is holding CAI back? One major factor is that the number of (pedagogic and computer) programming hours required, with present techniques, to build a CAI course, is too high. Generally accepted estimates give a range of 100 to 150 hours of programming per hour of lesson [4] [5].

It must be emphasized that the problem is not the cost of those programming hours, but the availability of personnel. Let us consider why.

In the majority of cases CAI is programmed instruction administered via computer. The programming is done frame by frame, by presentation of a concept, test of comprehension, analysis of answers to detect eventual misunderstandings, and provision of remedial frames or loops for every kind of error. An educational programmer must have a very deep understanding of the subject matter being taught, and must have enough teaching experience to predict the student's behavior. The computer programming side is really the least of the requirements and the trend has been to train the subject-expert in the use of the computer, and to develop ad hoc languages to make computer programming easier. More than 40 languages exist already for this purpose [6]. But this really solves the easiest part, because test, branching, and remedial loops have to be devised independently of the programming language, and it is there that the burden is located.*

In the state of the art, no subject specialist (unless his field is precisely teaching techniques) is willing to invest at least 100 hours in the preparation of

* The effort to develop a CAI system which explores its own data base [7] organized as a semantic net [8], in order to produce a natural language output is still in an experimental stage and, in spite of its theoretical value, it doesn't seem to solve the practical problem we are dealing with.
one hour of teaching. The higher his position, the truer this assertion.

Our system tries to solve this problem by eliminating the need for any kind of programming on the author's side. This means that the author doesn't have to worry about optimal sequences, or remedial loops, or computer languages, because the system includes algorithms for the organization of the educational material and for its presentation to students.

The system essentially consists of two computer programs.

The first program handles a logical description of the educational material prepared by the author, in order to build an optimal sequence, based on the pedagogical principles described in the following paragraph. The second program handles the presentation to each student in an adaptive mixed-initiative mode, i.e., the system tailors the presentation to the student's characteristics and assigns successive material according to: a) the structure developed by the first program b) the student's "level" c) the student's activity. At any stage the student may assume the initiative and make specific requests. Once those requests are satisfied the system regains the initiative.

2. PEDAGOGICAL STRATEGY

The following implementation characteristics are based upon generally accepted pedagogical principles.

a) The presentation sequence is built to optimize the contiguity of concepts between successively presented documents.

b) Extra rehearsal is provided each time that required concepts are forgotten.

c) Positive reinforcement on the one hand, and recycling with eventual additional material in case of mistakes, on the other, are provided as frequently as possible.

d) The system takes advantage of the teacher's experience by incorporating his introductory examples, definitions, proofs, comments and exercises as needed, and without sacrificing his personal style. The quality of the teaching material is probably more important than any other single element in this process.

e) The amount of material presented is related to each student's characteristics. A fast, abstract type learner receives a minimum of exposition, while the weaker student is provided with more introductory and complementary material. The structure of each course includes a "skeleton", comprising all the logically necessary parts
and the minimum additions to provide contiguity of concepts. For each student this skeleton is augmented with material that minimizes the expected time needed to complete the lesson, while attaining specified objectives. Based upon the student's actual performance, additional material may be presented to cover knowledge failures.

3. EDUCATIONAL MATERIAL

3.1 Definitions

In order to be precise, let us define our vocabulary.

We want to teach a course, and this will be done by a sequence of lessons, each one a session at the terminal.*

The teacher or author will be the person (or persons) in charge of the course, who writes the educational material.

The educational material will consist of a set \( D \) of documents, each one a written piece, coherent and purposeful. Examples could be: a definition, an exercise, a comment, etc.\(^\S\)

Every document establishes relations between certain concepts. Every one of those concepts belongs to one of three disjoint sets:

a) The set \( A \) of concepts to be taught.

b) The set \( B \) of prerequisites, which consists of concepts on which some of the \( A \) concepts are immediately based; the prerequisites must be mastered by the student in order to take the course.

c) The set of basic concepts assumed to be known by every student. They are, of course, a function of the environment (area of study, year, country, etc.).

In our system only \( A \) and \( B \) appear explicitly.

3.2 Documents organization

In the typical case the author will produce the set \( D \) of documents based on his class and research notes. These documents will be consecutively numbered, the order being unimportant, and we will indicate as \( d_i \) the document number \( i \).

* The teaching procedures are directly related to the hardware. We will assume that the terminal is able to transmit (both ways) character-string information.

\(^\S\) When the documents are presented on a display terminal they should fit into one screen presentation (or at most two).
He will also form a list $C$ of concepts, namely an enumeration of $A \cup B$. Let us call $c_j$ the $j$-th element of that list. The ordering of the list is arbitrary. The author must indicate which are the prerequisites, because this information is needed to organize the documents.

Our system has to build a teaching sequence for each student, using some or all of the documents in $D$. For this purpose we require additional information from the author. This information defines logically the documents from the system standpoint, and is divided in two classes:

a) A characterization of the documents regarding the function they satisfy with respect to the concepts they mention. We cannot know, by automatic means, whether a document contains a trivial comment or an indispensable definition. In order to make explicit the document-concept relation, we distinguish the following functions that a document may fulfill with respect to a concept.

- **I**: Introductory. Presentation of the concept.
- **N**: Necessary. Material that must be included, for logical or pedagogical reasons.
- **C**: Complement. Examples, applications, comments, etc.
- **E**: Exercise. Includes a question to be answered at the terminal; consequently it is associated with short response time.
- **P**: Problem. Demands time for thinking, development and/or operations. It would be uneconomical to keep a terminal idle during the relatively long period required for its solution. On the other hand, problems are necessary because they play an essential part in the development of student capabilities and in the deep testing of comprehension. Consequently, they will be assigned as homework.

Now, suppose a document is being used to teach certain concepts $a, b, c$. The functional relation of this document with respect to those concepts is described by means of the five indicators : $I, N, C, E, P$ just presented.

Nevertheless, the document may use other concepts $x, y, z$, already known by the student, in order to teach the concepts $a, b, c$, but without adding anything significant to the student's knowledge of $x, y, z$. In this case the concepts $x, y, z$ fulfill a function with respect to the document, and not the document with respect to these concepts. We will use the indicator $R$ to denote this type of document-concept relation.

This characterization of the documents is presented by the author as a matrix defined as follows: if the document $d_i$ mentions the concept $c_j$ the corresponding...
entry is I, N, C, E, P or R, according to the definitions just seen; if it does not, the entry is blank. When the document-concept relation may be qualified in more than one manner, the tie is broken according to the following priority sequence (high priority first):

E, P, N, I, C, R

b) The second class of information describes precedence relation between documents. For instance, if \( d_i \) contains the proof of a theorem which uses a lemma demonstrated in \( d_j \), then \( d_j \) must precede \( d_i \). This information is formalized by means of a binary precedence matrix. If \( i \) and \( j \) are document numbers, let us call \( e_{ij} \) the corresponding entry. If \( e_{ij} = 1 \), \( d_i \) must precede \( d_j \) in the exposition of the course. If \( e_{ij} = 0 \), the author did not establish explicitly a precedence relation between those documents. Usually the author does not write explicitly all the relations that may be deduced by applying transitivity. The transitive closure of the relation is computed by the system.

There is also a stronger precedence relation, which corresponds to the notion of successor. For instance, the sentence: "In the previous document we proved...", requires two documents to be consecutive. The author specifies this relation by means of "chains", i.e. sequences of documents to be included precisely as defined.

4. DOCUMENTS SEQUENCE

4.1 Conditions

The sequencing of documents must be decided by the system under the following priority-ordered constraints:

a) Precedence relations between documents.

b) Contiguity of concept areas for neighbouring documents.

c) Ordering of documents, that relate to the same concept, in a pedagogical series like I - N - C - E, or I - N - E - C - P, or similar, i.e. opening with introductory documents and closing with exercises or problems.

The first condition is the easiest to respect. The system receives a precedence matrix between documents and builds its transitive closure. The second and third conditions demand the definition of a "distance" between documents, which is presented in the next paragraph. The third condition is also dealt with by local reorderings.
4.2 A distance between documents

In order to introduce this concept, which is basic to the building of the sequence, let us start with an example.

With the notation of 4.1, suppose $\mathbf{A} = \{c_2, c_3, c_4\}$, $\mathbf{B} = \{c_1\}$, i.e. we have three concepts to be taught and one prerequisite. Let us assume also that $\mathbf{D}$ consists of 16 documents already sequenced by the author and numbered in the order of presentation. The corresponding matrix $\mathbf{DC}$ of documents vs. concepts is defined in Table 1; document $d_4$ is described by the $i$-th row.

<table>
<thead>
<tr>
<th>Con</th>
</tr>
</thead>
<tbody>
<tr>
<td>Doc</td>
</tr>
<tr>
<td>-----</td>
</tr>
<tr>
<td>1</td>
</tr>
<tr>
<td>2</td>
</tr>
<tr>
<td>3</td>
</tr>
<tr>
<td>4</td>
</tr>
<tr>
<td>5</td>
</tr>
<tr>
<td>6</td>
</tr>
<tr>
<td>7</td>
</tr>
<tr>
<td>8</td>
</tr>
<tr>
<td>9</td>
</tr>
<tr>
<td>10</td>
</tr>
<tr>
<td>11</td>
</tr>
<tr>
<td>12</td>
</tr>
<tr>
<td>13</td>
</tr>
<tr>
<td>14</td>
</tr>
<tr>
<td>15</td>
</tr>
<tr>
<td>16</td>
</tr>
</tbody>
</table>

Table 1. Matrix DC Documents vs. concepts

The sequence is satisfactory from a pedagogical standpoint. In effect, $d_1$ introduces $c_2$ in relation with the prerequisite $c_1$. Then, $d_2$ presents a necessary property of $c_2$, $d_3$ a complement and $d_4$ an exercise.

At $d_5$, which is necessary for $c_2$, $c_3$ is introduced. At $d_6$ we find a problem to test $c_2$ and, immediately after that, $d_7$ and $d_8$ describe $c_3$ based already upon the knowledge of $c_2$. The remainder of the sequence exhibits the same pedagogical characteristics.

If the author does not provide this sequence, we would like our program to build a similar one.

How is the "good pedagogical quality" of the sequence reflected in measurable parameters of the matrix? First, let us look at consecutive rows. The contiguity of concept areas, while introducing new material, is obtained by an overlapping of non-blank indicators. See, for instance, the transitions $d_4 - d_5$, $d_8 - d_9$ and $d_9 - d_{10}$.

Second, let us look at the table by columns, instead of doing it by rows. Each column describes the presentation sequence for its corresponding concept. The standard column is a sequence of interwoven clusters, each of which corresponds to one of the relational indicators. The typical sequence of cluster indicators is:
but we cannot force this order upon each column because of the row linkages.

In order to satisfy these competitive requirements as much as possible we create a measure to evaluate sequences of documents. The main idea is to define a distance between documents, and to organize the sequence to minimize the average distance between successive documents.

We start with a distance between relational indicators, and we want this distance to reflect the typical sequence of clusters indicators, i.e. the distance must be minimal for consecutive indicators and then it must grow according to the separation in that sequence.

A linear assignment is excluded because of the presence of the same indicator at both ends of the sequence. Our solution, which encompasses the whole set of requirements, is to map the set of relational indicators into points in a plane, as shown in figure 1, and to define the distance between two indicators as the distance between the correspondingly labeled points.

To see the properties of this distance, let us consider the distance between the k-th elements of documents $d_i$ and $d_j$.

The distance between $DC_{i,k}$ and $DC_{j,k}$ will be (relatively) small when both
documents mention \( c_k \), and (relatively) big when one of the documents mention \( c_k \) while the other doesn't (blank character). When both \( d_i \) and \( d_j \) mention \( c_k \), the distance corresponds with the sequence of cluster indicators. Of course, if \( DC_{i,k} = DC_{j,k} \), the distance between these elements is zero.

Now, as a normalized distance between vectors \( \vec{d_i} \) and \( \vec{d_j} \) we will use the expression:

\[
\frac{|\vec{d_i} - \vec{d_j}|^2}{|\vec{d_i}|^2 + |\vec{d_j}|^2}
\]

i.e. calling \((x_{ij}, y_{ij})\) the cartesian coordinates that correspond to \( DC_{i,j} \), the distance between \( d_i \) and \( d_j \) is given by:

\[
\text{DIST}_{i,j} = \frac{\sum_{k=1}^{n} \left( (x_{ik} - x_{jk})^2 + (y_{ik} - y_{jk})^2 \right)}{\sum_{k=1}^{n} \left( x_{ik}^2 + y_{ik}^2 + x_{jk}^2 + y_{jk}^2 \right)}
\]

where \( n \) = number of concepts.

This normalized distance reaches its minimum (0) when both vectors are equal, and its maximum (1), when each time that a concept is mentioned in one document, it is not mentioned in the other.

**Note:** Clearly, there is a whole family of expressions that satisfy the same constraints. This assignment of points in the cartesian axis and the expression used gave us the best results in a set of different trials. The measure of success in the trials was given by the number of inversions of the computer-produced sequence with respect to the author-produced sequence. Of course, there is a subjective component, and one of the lines of research open is precisely how to optimize, in this context, the definition of distance between documents.

The system actually uses a measure of similarity \( SD_{i,j} \) between documents \( d_i \) and \( d_j \) defined by:

\[
SD_{i,j} = 1 - \text{DIST}_{i,j}
\]

Maximum similarity (1) is obtained for documents that have the same row representation, and minimum similarity (0) for documents that mention entirely different concepts.
4.3 Building the sequence

The presentation sequence is built in two stages: first, all precedence related documents are organized and, second, the remaining documents are inserted.

The first sequence is built step by step, starting with the document without predecessors that is most similar to an ad hoc vector of prerequisites, and adding, each time, one of the documents that has no predecessors in the set of non-yet-sequenced documents. The selection of the document to add takes into consideration its average similarity to the documents already sequenced (attraction), its average similarity to the non-yet-sequenced documents (repulsion) and its similarity to the last sequenced document (local attraction). When possible, this first sequence is locally reordered by advancing the introductory documents.

In the second stage the remaining documents are inserted, one by one, in a three-step process:

a) selection of the document (attraction and repulsion)

b) determination of the insertion area (area of maximum average similarity to the document)

c) effective insertion (local decision).

5. THE TEACHING SYSTEM

5.1 General organization

In order to take a course each student is subjected to two tests of a different kind: a general one to determine his "level"*, and a specialized one to establish his knowledge of the course prerequisites. If the results of the specialized test are acceptable, the student's level determines the particular sequence of documents to be presented. This sequence should be the "best" we are able to produce with the knowledge gathered up to that moment.

* How to properly assign a "student level", i.e. how to map a complex set of intellectual characteristics into one dimension, will not be discussed in this paper.
We may visualize this sequence in Figure 2:

![Diagram of sequence of documents]

Each node of type \( s \) or \( c \) consists of one document or a sequence of documents that must be presented consecutively. The circular nodes indicate the "skeleton" mentioned in 2.e; i.e. those documents that we consider necessary even for the brightest and most abstract-type student. The square nodes denote the documents that, added to the \( s \)'s, will provide the sequence considered optimal for the particular student. The triangular nodes contain the auxiliary material, grouped in such a way that each \( r_i \) may supplement the corresponding \( c_i \).

Note: At this point an underlying assumption must be emphasized. The student population is presumed to be a "class", i.e. a relatively homogeneous set of students. Furthermore, the author is familiar with the general characteristics of the class, and this is reflected in the material he produces, i.e. the most difficult document is accompanied by enough introductory or complementary documents so as to make it comprehensible by the weakest student allowed to take the course.

Because of this assumption the author does not prepare different sequences for different student levels. He just adds explanations to his basic material.

5.2 Expository documents

When the student finishes reading an expository document (not a problem or an exercise) he types a character whose meaning and effect are summarized in the following table:
Luis Osin

<table>
<thead>
<tr>
<th>Character</th>
<th>Meaning</th>
<th>System action</th>
</tr>
</thead>
<tbody>
<tr>
<td>U</td>
<td>understood</td>
<td>Introduction of the next document in the sequence the student is presently following.</td>
</tr>
<tr>
<td>C</td>
<td>clarify</td>
<td>Organization of a remedial loop, beginning with an already understood document, introducing new material, and finishing with the document whose clarification was asked.</td>
</tr>
<tr>
<td>B</td>
<td>back</td>
<td>Presentation of the previous document. Successive B's take the student back in the sequence he followed.</td>
</tr>
<tr>
<td>S</td>
<td>suspend</td>
<td>The session is closed. The student will resume at the same point when starting a new session.</td>
</tr>
<tr>
<td>?</td>
<td>doubts about certain concepts</td>
<td>Upon receiving the list of concepts to review, a sequence is presented covering all the necessary documents related to those concepts up to the document where the question was raised.</td>
</tr>
<tr>
<td>R</td>
<td>resume</td>
<td>Return to the original sequence. If, at document $d_i$, the student starts a chain of remedial processes (B, C, ?) because of a doubt, he may go back to $d_i$ at any point of the chain by typing R.</td>
</tr>
</tbody>
</table>

5.3 Exercises and problems

The objective of a problem or exercise, in this context, is to test the understanding of certain concepts being taught. Being so, the classical dichotomy of "right" or "wrong" is useless here. It is clear that a problem with a wrong solution may correspond to a student who understands those concepts (and has made a numerical error). Alternatively a correct final answer may correspond to a student who didn't understand the necessity of proving essential steps.

Another aspect worth mentioning is that in most CAI systems the student's answer to each exercise gives the only decision criteria to govern branching. Our system adds other branching criteria based on a variety of student responses (B, C, R, ?). This means that we don't need an exercise or a question after each document. Nevertheless, the student may think that he understands concepts while he doesn't, so that objective tests are indeed necessary.

At this stage the system has been programmed to handle two types of exercises:

a) Multiple-choice

The author specifies the correct choice and, for each incorrect choice, he
indicates the concepts misunderstood. When the student answers correctly he receives explicit approval and the next document in his sequence is presented. When the answer is incorrect the system response may vary. If it is the first presentation of that exercise, the remedial loop includes at least those necessary documents that are logical predecessors of the exercise. A minimal second presentation contains all the necessary documents and if the student fails again a still richer sequence is built.

b) Free (short) answer. The logical structure is the same as in a), but in this case the author adds a list of possible (short) answers which are mapped onto a correct choice and several incorrect choices (each one corresponding to a list of conceptual failures). The student's answer is preprocessed to find the corresponding choice and then the system proceeds as in a).

Note: Other possibilities have been successfully implemented in existing CAI systems and some of them will be (hopefully) added to this system, but it must be emphasized that it will always be with the same philosophy: the author does not design remedial loops. He indicates the conceptual failures corresponding to the student's possible answers, but the system takes care of the correcting steps, according to the student's level and activity.

The correction of homework problems is done by assistants who return, for each incorrectly solved problem, the list of the student's conceptual failures. The system designs remedial loops with criteria similar to those previously described for exercises. The system also checks the delay in the reception of homework assignments and may block the student's activity if his grades are not received within a certain period.

6. THE AUTHOR'S ACTIVITY

We shall sketch the development of a course as a non-deterministic program. The author must follow the instructions in the established order, unless a possible branching is specified.

1. Specify the set A of concepts to be taught and the set B of prerequisites, as defined in 3.1.

2. Write, correct, or collect a set of documents in order to teach A. If necessary, go back to 1.

3. Have the set of documents, or part of it, read by your assistants and friendly colleagues. If you think that some of the criticisms are worthwhile, go to 2.
4. Build the D vs. C matrix defined in 3.2.a. This is done by rows. Now look at it by columns and probably you will find that not enough adequate material was prepared for certain concepts. If this is the case, go to 2. Specify the prerequisites.

5. Build the D vs. D matrix defined in 3.2.b. Specify "chains", if necessary.

6. Input the data prepared in 4 and 5. The system will print the sequence of document numbers described in 4.3. Alternatively, the system will indicate the existence of incompatible precedence relations and it will print them.

7. In case of incompatibility, make the appropriate corrections and go to 5 or 2.

8. Arrange the documents according to the sequence obtained in 6 and read them. If you are not satisfied with the sequence, go to 2 or to 5.

9. Add, to the system produced sequence, a new dimension called document level. The set of level 0 documents constitutes the previously described "skeleton". All the necessary (N) documents receive level 0, but you are free to assign this level to any other document that, for pedagogical reasons, should be read by every student. Now, assign level 1 to the documents you consider as the best addition to be read by the not-so-brilliant students. Continue with levels 2, 3, etc., up to the exhaustion of the documents. The maximum level that may be assigned, in the actual implementation, is 8, but usually not more than four of five levels are required.

10. Teach the course, using the system, to a small group of selected students. They must be responsible students, capable of enjoying the novelty and willing to make the effort. As an incentive, and a well merited compensation, the usual credit units should be assigned for completing the course. According to your evaluation of each student (his average grade in related subjects could be a basic measure) assign to him a "student level" number (an integer between 0 and one less than your maximum document level). The system will present to any student of level i all the documents of level j, 0 ≤ j ≤ i. In other words, referring to figure 2, the s-documents are those of level 0, and the c-documents for a level i student are those whose level ranges from 1 to i. The higher level documents are used for remedial loops. The student level may be dynamically adjusted by the system according to his performance.

11. On completion of the course you will receive the complete history of each student, and the statistics of each document.
a) History - The path (sequence of steps) followed by the student will be described as a sequence of four-tuples: \((i, j, t_{ij}, a_{ij})\) where:

- \(i\) = ordinal number of the step
- \(j\) = number of the document read
- \(t_{ij}\) = time required to read document \(j\), in step \(i\)
- \(a_{ij}\) = answer (U, e, etc.)

In the case of problems, \(a_{ij}\) will be the grade received.

b) Statistics - For each document \(d_j\) a seven-tuple \((j, f, s, tf, ts, cf, cs)\) will be printed, where:

- \(j\) = number of the document
- \(f\) = number of first presentations (coincides with the total number of students who read it)
- \(s\) = number of additional presentations
- \(tf\) = average reading time on first presentations
- \(ts\) = " " on later
- \(cf\) = average comprehension on first presentations (percentage of U-answers)
- \(cs\) = average comprehension on later presentations

12. Analyze the statistics obtained in 11, and correct documents accordingly. Excessive average reading time or very low average comprehension will prompt you to polish or rewrite a document. The analysis of students' paths will guide you to correct either document levels or the assignment of levels to students.

13. Go through the sequence of documents with some of the students that took the course and take advantage of their opinions, which originates from natural experience. Their difficulties will show you the need for new material or for rewriting or even excluding some of the existing documents.

14. According to the changes introduced in 12 and 13, modify the structural information that defines the course and perform 6.

15. Adjust the criteria for assignment of student levels.

16. Teach the course, using the system, on a production basis.

17. Go to 11. With time, corrections will diminish, but new documents may be added.
7. EVALUATION

Until now the system has only been tested on short courses. In these cases the author had no problem in producing the required matrices. For a long course the generation of large matrices can be avoided by dividing the course into chapters, each one of which is a course from the system standpoint.

An encouraging result is that the time spent by authors using this system vs. authors using "classical" CAI is reduced by one to two orders of magnitude. The organization of, let us say, twenty five documents, which cover an average one-hour lesson, takes about two hours of author's time.

The efficiency of the system in its relation with students has not really been checked. One of our next activities will be to teach one course to a group using this CAI system and to compare the results with a matched group receiving classical instruction.

Of course, the system described is not a universal panacea for education, not even for CAI. Educational procedures must include classical lecture, laboratory, small-group study, independent work, programmed-instruction, audiovisual techniques, CAI in different forms, and every reasonable way of conveying information to students. Which combination of these procedures has to be chosen depends on the optimization of a weighted set of parameters including cost, availability, students' and teachers' time, and students' and teachers' satisfaction. Up to now CAI could not really compete with the other procedures because of its very restricted availability. This system attempts to lift that restriction.
8. REFERENCES


[6] Zinn, K. "Requirements for programming languages in computer-based instructional systems", in [9].


[8] Quillian, M. Ross "Semantic Memory", in [8].
