Order Number 9119654

An exploratory educational environment for computer-supported education

Masullo, Miriam J. Salman, Ph.D.

City University of New York, 1991

U·M·I
300 N. Zeeb Rd.
Ann Arbor, MI 48106

A

An Exploratory Educational Environment for Computer Supported Education

by

Miriam J. Salman Masullo

A dissertation submitted to the Graduate Faculty in Computer Science in partial fulfillment of the requirements for the degree of Doctor of Philosophy, The City University of New York.

1991

This manuscript has been read and accepted for the Graduate Faculty in Computer Science in satisfaction of the dissertation requirement for the degree of Doctor of Philosophy.

1/21/91
Date

Michael Anshel
Chair of Examining Committee

1/23/91
Date

K. Wendroff Heegaard
Executive Officer

Dr. Robert Nirenberg

Dr. Shirley Feldmann

Dr. Seraphin Calo

Supervisory Committee

The City University of New York

Abstract

Exploratory Educational Environment

by

Miriam J. Salman Masullo

Adviser: Dr. Robert Nirenberg

Ascertaining the need to combine Computer Science with Education and Psychology in order to meet the challenges that will close this century, this work presents the **Exploratory Educational Environment (E^3)**, an infrastructure for **Computer Supported Education (CSE)** that employs a knowledge-based approach for the widespread use of computers in Education, and facilitates the automatic production of **Individual Educational Programs (IEPs)** targeted to meeting the individual needs of each learner.

The E^3 Model consists of:

(1) **an open systems architecture** specially designed to support functionally organized knowledge and information coming from many sources;

(2) **a learner model** meant to capture and record the learning process; and,

(3) **an instructional modulation and software adaptability scheme** for discriminating among different alternatives and amplifying the power of instructional software.

This research suggests that attempts being made by the educational community to individualize instruction can indeed be facilitated by computer technology. However, in order to be able to coordinate and organize the new opportunities for learning created by the use of such technology, an infrastructure is needed to bring together not only the new and needed contributions, but also the traditional elements that play a role in the educational process. Conceptually, E^3 is such an infrastructure that can be used to characterize the learners and the instructionals matching them more systematically.

It is further suggested that E^3 has been designed in such a way that it can be used to exploit both existing and emerging computer technology, and can therefore serve as a stable, long term platform for the application of computers to education.

The overall goal of this work is thus to bridge the gap that exists between Education and Computer Science.

Acknowledgements

Technical guidance concerning Computer Science issues was provided by Dr. Robert Nirenberg and Dr. Michael Anshel, The City College and The Graduate School and University Center of The City University of New York.

Guidance and review of all Educational issues was provided by Dr. Shirley Feldmann, Department of Educational Psychology, The City College and The Graduate School and University Center of The City University of New York.

Technical support concerning Systems Management issues as they related to Educational Systems Management was provided by Dr. Seraphin Calo, Systems Management Technologies, Thomas J. Watson Research Center, IBM Research Division, Yorktown Heights, New York.

Encouragement and support for the interdisciplinary nature of this research was provided by Dr. Thomas Wesselkamper, Department of Computer Science, The Graduate School and University Center of The City University of New York.

Many technical comments concerning Computer Science issues were provided by Dr. Joseph Hellerstein, Systems Design and Analysis, Thomas J. Watson Research Center, IBM Research Division, Yorktown Heights, New York.

Much general information and reading materials about current educational practices and issues were provided Dr. Linda A. Tsantis, and David Keefe, Special Education Programs, IBM Educational Systems, Rockville, Maryland.

The information, references and insight necessary to create the scenario for E^3 Mathematics were provided by Dr. Robert Gyles, Director of Math, Science and Technology for Community School District 4 in East Harlem New York, and by Gale Rubenstein, Director of Computer Education for Community School District 4.

Editing and the formulation of target dates for completion were the work of Robert Ennis, Systems Management Technologies, Thomas J. Watson Research Center, IBM Research Division, Yorktown Heights, New York.

Text processing assistance during the creation of this document was provided by Elizabeth McCarthy, Computing Systems Department, Thomas J. Watson Research Center, IBM Research Division, Yorktown Heights, New York.

Personal motivation for this work was provided by Diane and Joey.

This research was supported by Thomas J. Watson Research Center, IBM Research Division, Yorktown Heights, New York.

this research is dedicated to all

future caretakers of this earth

Contents

Figures

Part One: Introduction

Section 1. Statement of Purpose

The problems that our Educational System face need not be compounded with uncertain computer-based solutions. It may not be utilitarian to throw away a system put into production via a design and development process still progress, after more than a century. And specially when there is no analog in Computer Science to effectively deal with such design and development impasse.

Fundamentally, this research points out that our Educational System can coexist with the new technology and reap all of its benefits. That it can do so in its own terms, without having to adapt to approaches that revisit the way in which children have been known to learn, and educators have learned to teach. This is one avenue seldom explored, because it is conservative in approach. The pages that follow are an effort to employ computer technology for the enhancement and demonstration of educational practice and its possible evolution.

What is put forth is an organizing structure that is formal, to support a process that is ad-hoc and may be becoming even unconventional. While the end product may very well employ drastically new approaches, the mechanism that computes, creates and delivers it need not be unpredictable.

In doing so the few traditions of the newer discipline are endorsed as well. Since computers were introduced in many areas of industry, applications have been developed by a process of studying the human-based approach and mechanizing

it gradually, but formally, towards massive applicability. That has been known to work. It will probably also work in the area of Education.

On the other hand, it is widely accepted that technology must lend a helping hand where traditional educational systems are failing, that includes both the classroom and the home as sources of learning. However, what is not widely accepted is the fact that in order to use technology one must be technical. Terms such as user-friendly and personal computers disguise the fact that computers are not eminently meant for people.

It seems highly incongruous to have to be comfortable with abilities particular to computers scientists and engineers, in order to be able to use the same technology as a tool for learning Kindergarten fundamentals and other things of lasting value. The challenge is, and has always been, to formulate the kind of infrastructure that facilitates the use of computer technology for educational purposes.

Such infrastructures exists in other areas of technology. For instance while telephone networks have been referred to as the largest computers in the world, few of its many proficient users even know that. Another good example is the television. The day has not arrived yet when computers can be plugged in and turned on to render a service. It didn't take industry long to discern that people were not going to design, film and broadcast material in order to be able to use

televisions. Therefore a complex infrastructure has been put in place to facilitate use of the media.

Since computers are in general more potentially serviceable that television sets, a number of specialized infrastructures are needed. For the use of computers in Education this concept of a complex, open and recipient-targeted infrastructure does not yet exist, had not been formulated, and until it is put in place computer technology will not be meaningfully and massively applied to Education.

Section 2. Necessary Information

The language used in the exposition of this research was kept mostly outside the technical jargon of computer programming. Some Computer Science terminology was needed, in particular the use of Artificial Intelligence (AI) terms. In this section an overview of the area covering only those issues that relate to this research is given for the benefit of the interdisciplinary audience. In addition a list of terms is provided.

Knowledge-Based Systems Overview

In this document the term **knowledge-based system** is used in the context of **Artificial Intelligence**. Such a system is an application by computer, of human knowledge represented in some way in computer programs. These systems also require some form of inferencing mechanics, such as an **inference engine**.

An inference engine is a computer program. It is the program or piece of software that controls the activities of the knowledge-based system. Typically some form of **matching** is involved. For instance **conditions** are compared (matched) with incoming information (data). When enough data arrives to **satisfy** a condition (all of the requirements are met) then specific sets of actions are executed.

Thus a knowledge-based system is a program that consists mainly of **condition/action pairs**. These are commonly known as **rules**. Rules then are

written as **left hand side (LHS)**, or condition and right hand side (RHS) or action pairs. The arriving data is organized in what is referred to as **working memory**. This is a form of database where elements of that data are stored during the inferencing process.

When the LHS of more than one rule is satisfied, then more than one possible course of action can be taken. In those cases the inference engine must select a single rule and trigger the execution of its RHS. This is done by mean of a process called **conflict resolution**. The set of all candidate rules is referred to as the **conflict set**.

The rules themselves are also referred to as memory, sometimes called **production memory**. A knowledge-based system which uses rules as its form of knowledge representation is also known as a **production system.**

More importantly the general programming style exhibited in the knowledge-based approach has sometimes been described as **declarative**. The term is often defined as opposed to the more traditional **procedural** programming style. In the procedural approach the programmer controls by design and implementation the order in which processes are carried out by the computer. The declarative approach delegates those decision to a program over which there is no explicit form of control. In the context of this discussion that would be the inference engine.

Recent developments in the area of knowledge-based systems have provided facilities and tools through which it is possible to have some form of both explicit and implicit control over processes. Programming in a combination of both the procedural and declarative styles offers many otherwise unavailable design opportunities and may lead to solutions that are more comprehensive than if implemented in either one paradigm alone.

Section 3. List of Terms

A

access. The manner in which files or datasets are referred to by a computer.

access mode. A technique that is used to obtain a specific logical record from, or to place a specific logical record into, a file.

active program. Any program that is loaded into memory and ready to be executed.

activate (a block). To initiate the execution of a block.

activity. The percentage of records in a file that are processed in a run.

algorithm. A finite number of well defined steps for the solution of a problem. Contrast with heuristic.

allocate. To assign a resource for use in performing a specific task.

analysis. The methodical investigation of a problem, and the separation of the problem into smaller related units for further study.

application expert. A person who is a source of expertise for solving problems in a specific subject area. This person may know the subject well but may not have advanced programming skills. If the application expert has limited programming experience, he or she supplies information to the knowledge engineer, who creates a knowledge-base and knowledge application. If the application expert has enough programming experience, he or she may be able to enter data into the knowledge-base without the help of a knowledge engineer.

application user. A person who queries a knowledge application in a consultation environment to solve a specific problem. Also a person for whom the knowledge application provides a service or facility. Also called end user or client.

artificial intelligence. The capability of a device to perform functions that are associated with human intelligence.

asynchronous. Without regular time relationship; unexpected or unpredictable with respect to the execution of a program's instructions.

attach. To create a task that can be executed asynchronously with the execution of the mainline code.

attribute. A characteristic; for example, attributes of data may include record length, record format, associated device information. Also a descriptive prop-

erty associated with characteristics or properties used to described characteristics.

auditory perception. The ability to receive and understand sounds.

auditory reception. The ability to understand the spoken word.

automatic. Pertaining to a process or device that, under specified conditions, functions without human intervention.

automatic programming. The process of using a computer to prepare computer instruction code.

automation. The implementation of processes by automatic means. Also the conversion of a process to automatic operation. The investigation, design, development, and application of methods for rendering processes automatic.

auxiliary equipment. Equipment not under direct control of the processing unit.

auxiliary storage. A storage device that is not main storage. Storage that supplements other storage.

availability. The degree to which a system or resource is ready when needed for processing.

B

block. A delimited sequence of statements used to define related operations to be performed on specified data.

block structure. A hierarchy of program blocks.

BOCES. Board of Cooperative Educational Services, computer supported educational management services provided by the State of New York to its educational districts.

C

class. A data type used to declare working memory.

class attribute. A property of a class, such as color, age price or size. Class attributes are associated with values for specific class members; for example, the value of color may be red. You may also think of class attribute as a field in a record, or a column in a table.

class member. A single member of a class, like a row in a table.

cognitive structure. The mental processes by which an individual becomes aware of and maintains contact with his internal and external environments.

cognitive style. An individual's characteristic approach to problem solving and cognitive tasks.

condition. Specifies the logical conditions for class members in working memory. Conditions are included in patterns.

conflict resolution. The process by which a rule whose conditions are satisfied by class members in working memory is chosen to be executed.

conflict set. The group of instantiations that contain rules eligible to fire.

committee on special education (CSE). A team of people selected to assess and plan to match special educational needs with special educational programs.

computer supported education (CSE). An integrated approach for computer-based educational systems..

D

data-driven. When data causes a production system to execute. Also called forward chaining.

declarative programming. A programming style or technique in which facts, rules, and assertions are presented by relating them to each other. The order in which statements are processed is not predetermined; instead, the relationships

influence the processing. For example, in rule-based programs, rules can be fired in any sequence, depending on the data elements found to be true at any given time. The inference engine dynamically creates the control structure. Contrast with procedural.

delivery environment. The set of product characteristics seen and used when an application is put into production.

development environment. The set of product characteristics seen and used by the programmer who develops and tests the application. This environment may contain additional tools, such as debugging tools, tracing tools and hardware that are different from those in the actual delivery environment.

domain. An application area of knowledge.

E

end user. See application user.

event. A request to send a class member to the working memory of an application at a certain time.

event-driven. A software action will be taken when the data is changed.

expert system. See knowledge-based system.

F

fire. To execute the statements or actions on the right-hand side of a rule.

forward chaining. An inferencing mechanism that starts with available information, and draws conclusions from it that are appropriate to the goal.

H

heuristic. Pertaining to exploratory methods of problem solving in which solutions are discovered by evaluation of the progress made toward the final result. Contrast with algorithm.

I

individual educational program (IEP). An educational program planned to meet individual educational needs.

individualized instruction. A program planned to meet individual needs.

inference engine. The mechanism used to draw conclusions based on the rules in the knowledge-base and the data for the problem. The inference engine is a reasoning mechanism by which new facts are driven from known facts. Infer-

ence engines can use backward chaining or forward chaining, among other techniques.

inferential reasoning. The act of obtaining a judgement or logical conclusion from given data or premises.

instantiation. A rule and the list of class members that satisfies the left-hand side of that rule.

integrated learning systems. A computer-based instructional system defined in terms both hardware and software.

K

knowledge acquisition. The transfer and transformation of problem solving expertise from some knowledge source to a program.

knowledge application. A specific application of a knowledge-based system, including a knowledge-base created by a knowledge engineer or application expert. Such an application is ready to be used by application users.

knowledge-base. A collection of problem-solving information expressed in a knowledge representation language. Usually this information is in three parts:

rules (conditions and actions), constraints, and data elements (that is, class members with attributes and values).

knowledge-based system. A computer program, or group of programs, using specific knowledge in a narrow problem area and requiring complex inferential reasoning to solve problems that require human expertise for their solution. Knowledge necessary to perform at such a level, combined with the inference procedures used, can be thought as a model of the expertise of the practitioner of the field.

knowledge engineer. A person who creates a knowledge-base containing the problem-solving information of an application expert as part of a knowledge-based application.

knowledge engineering. The process by which a knowledge-based system is developed. A knowledge engineer observes and interviews a domain expert to identify and encode the expert's knowledge.

L

left-hand side (LHS). The part of a rule that specifies patterns to match against class members in working memory. Also called condition.

linked list. A data structure composed of data elements that relate to other data elements in the structure by means of pointers.

M

match. To evaluate the conditions of rules to determine which are satisfied given the current contents of working memory.

N

negative patterns. Specifies that no class members satisfy the pattern.

P

patterns. The part of a rule's left-hand side that tests for the existence of a class member and, optionally, states the conditions the class member must satisfy.

pattern matching. The process that compares class members against patterns to determine the conflict set.

PLATO. Program Logic for Automated Teaching, computer-based teaching system.

pointer. An identifier that indicates the location of a data element.

priority tag. A symbol defined to indicate a rule's priority for firing.

prescription. Activities utilizing diagnostic information for students with special needs.

procedural programming. The traditional form of programs, in which statements are structured sequentially. The order in which operations are to be performed is made explicit when the program is created. FORTRAN, PL/1, and C programs are examples of procedural code. Contrast with declarative.

procedure-block. Also called procedure.

procedure. A course of action taken to solve a problem. A collection of statements, often headed by a Procedure statement and ended by and End statement. These may be called from the right-hand side.

production. A condition action statement. In some languages *if then else* statements. Productions can also take the form of *When condition Begin action*. See also Rule.

production system. A program composed entirely of conditional statements called productions. See also knowledge-based system.

production memory. The rules in a knowledge-based or production system.

R

realtime. Multiple software activities proceeding asynchronously. The term is also used to describe systems operating in conversational mode and processes that can be influenced by human intervention while they are in progress.

remediation. That function which redirects or circumvents an impaired procedure in learning. It implies compensatory methods which facilitate learning.

recognize-act cycle. The iterative cycle of events in a production system. A loop in which rules with satisfied antecedents (LHS) are found by the inference engine, one is selected, and its actions (RHS) performed.

RETE. An efficient pattern matching algorithm for production systems.

right-hand side (RHS). The action part of a rule.

Rule. Commonly what the knowledge engineer receives from the expert during the interview part of the knowledge acquisition process. Also a condition action pair. This term is sometimes used interchangeably with production.

rule firing. When the inference engine by means of conflict resolution selects one production with a satisfied left-hand side, and performs the actions specified in the right-hand side; that is it fires the rule.

S

Sieve of Eratosthenes. A process to help filter out the composite numbers leaving only the prime numbers.

structuring. The act of arranging an activity in a way that is understandable and conducive to performance.

subdomain. A natural subdivision of a problem.

subprocedure. A rule subroutine.

T

time tag. A unique identifier for a class member that represents its recency in relation to other class members.

V

verbal expression. The ability to express ideas in spoken words.

visual association. The ability to relate visual symbols in a meaningful way.

visual reception. The ability to comprehend pictures and written words.

W

when. A rule statement. The left-hand side specifies conditions under which the actions on the right-hand side should be performed.

working memory. The main memory of a knowledge-based system. A global database in main memory, typically containing information about the state of the problem being solved.

Section 4. Organization of this Document

This document is organized as follows.

Part One - Introduction

The introductory chapter contains the following information:

- a statement of purpose,
- a definition of terms used,
- a brief description of the organization of the document.

Part Two - Background

This is research background material including the following:

- a discussion of the research problem,
- a review of the state-of-the-art,
- the human-based approach selected (presented in some detail, thus establishing the grounds for transition to a computer-based implementation).

Part Three - Architecture

The architecture for the proposed solution is presented as follows:

- the human based strategies are mapped to the computer design approach,
- a system architecture is introduced specifically to support this approach.

Part Four - Methodology

Specific issues in the architecture and instructional organization approach are discussed.

1. With respect to the architecture the following issues are discussed:
    - isolation of functionality,
    - knowledge representation,
    - and procedural knowledge.
2. With respect to the instructional organization the following issues are discussed:
    - modeling,
    - software adaptability.

Part Six - Summary

Issues relating to this research which were not formally addressed are discussed. These include the following:

- related computer science issues,
- preliminary implementation suggestions,
- reaching the classrooms.

There is also a summary of research contributions.

Part Two: Background

Section 5. Research Problem

The educational needs of the American people are not being met. Much has been been published on this topic but the prominently motivational agent for reform is *A Nation at Risk: The Imperative for Educational Reform*, The National Commission on Excellence in Education, 1983, which was responsible for making the nation more widely aware of what is considered to be a crisis in national education.

Subsequent reports confirmed the findings: *The Action for Excellence* report of the Task Force on Education for Economic Growth by the Education Commission of the States, the *Report of the Twentieth Century Fund Task Force on Federal Elementary and Secondary Education Policy*, and *America's Competitive Challenge: The Need for a National Response*, of the Business Higher Education Forum. All these documents tend to agree that while the nation at large was the target of information on this trend of educational suicide, the educational community could not be surprised, (Senese, 83).

In an effort to find ways in which to meet the educational needs of the American people, educators have explored many new forms of technology. *Teaching Machines, Learning Machines, Computer Aided Instruction, Computer Assisted Instruction, Computer Managed Instruction* and other such hopes enticed the communities (Education, Psychology and Computer Science) into believing that further evolution of educational practice must strive in the application of com-

puter technology. However, the use of the technology has not resulted in (appreciable) improvements in learning (Clark, 85). And after three decades of study and practice, the use of computer-based instruction remains largely an issue of exploration. The number of Ph.D. dissertations published each year dedicated exclusively to the assessment of usefulness and results of the technology, proves that the communities are still both greatly concerned and bewildered.

Specific Issues

In *Assessing the Impact of Computer-Based Instruction*, (Roblyer and Castine 88) the reason is made clear: there are too many open questions.

Summarizing some of those questions.

- Can computer applications help improve performance in basic skills and other key areas?
- For what specific areas, grade levels, and content areas are computer applications most effective?
- Which kinds and levels of students seem to profit most from using computers to learn?
- Which kinds of computer applications are most effective, for which skills and content areas?

- Can computer applications improve students' attitudes toward school, learning, and their abilities to learn?

- Will improved attitudes translate into better performance in school and lower dropout rates?

After three decades "the answers to those questions remain largely unanswered - and the questions themselves often unasked". Educators in general complain that existing technologies have not yet been meaningfully applied to Education. Therefore, expectations from emerging technologies are not realistic since there is no existing infrastructure for their use.

Failure in the use of the technology has been attributed to lack of integration with learning theories (Petkovich and Tennyson, 84,85). But the first order of integration, providing the ways and means for integration has not been taken up formally. Even more fundamentally, educators don't know what is available in educational software, due to the incoherent proliferation of tools, and even if they did know, there is no organized way of matching what is available to the students' needs.

Section 6. Introduction to the Proposed Solution

The Computer Science community has a responsibility to help resolve these problems. Those who have taken up the challenge extrapolate various aspects for research. It is seldom the case, given the proliferation of computer systems and computer techniques available, that a single major work attempts to simultaneously address all of the problems.

In this research tools are presented for simultaneous explorations of the more significant problems. This work will help establish harmony and consistency in the ways and means of exploration, and in the way of gathering empirical data that can then be used for studies that help answer questions and solve problems. One of the main goals is to create a vehicle whereby the related disciplines can work together in a concerted effort to provide a set of comprehensive solutions. **It is in fact the main stated goal of this work to bridge the gap between Education and Computer Science.**

Primary Objectives

This research expands the ways in which advanced computer technology can be used to provide coordination of educational services among the three major areas for educational computing:

1. administration (multipurpose tracking mechanism),

2. management (managing educational goals), and
3. instruction (providing instructional tools).

In doing so a great portion of the resulting integrative contribution is to facilitate the study of the problems and questions posed previously, for education researchers at large. And further ways in which this coordination of services can be provided across the three main arenas for learning:

1. the family/home,
2. the classroom, and
3. the tutoring environment.

Fundamental Requirements

Some of the fundamental requirements for this system follow.

- Adapt education to students' needs:
  - reducing stereotyping of students by levels and labels;
  - providing individual education for each student, regardless of needs, strengths or patterns of learning preference;
  - making the situation ready for the students, instead of the students ready for the situation.

- Enhance the value of educational tools:
  - encouraging the design and development of educational tools;
  - coordinating the use of many different educational tools;

- making combined educational tools more valuable than the sum of their parts.

- Make particular instructional tools that run in the environment integrate education-related functions:
  - tracking the student dynamically:
    - tracking long-term progress of the student,
    - tracking short-term progress (within sessions);
  - planning and constructing sessions according to current student needs:
    - dynamically adjusting session plans to maintain student interest and progress,
    - relieving monotony by switching sets to add the unexpected.

As a result of meeting these requirements further research and development in the general areas affected will be simplified and promoted and many avenues for specialized and empirical studies will be opened.

Section 7. Areas of Needed Research

The report by Roblyer and Castine (88) is presented as a form of evidence that the proposed solution is in fact needed. Open areas of research as summarized in that report span across specific factors. While those are not solely the basis for this research, they have nevertheless been given strong consideration and suggestive ways in which to deal with them within the E^3 approach are provided.

- Open questions relating to applications in various skills and content areas are formulated as follows:
  - measuring effects of a specific kind of application designed to teach one type of mathematics or reading skill,
  - comparing effects of a specific kind of application designed to teach reading with one or more strategies,
  - comparing effects of similar programs from the same developer,
  - measuring effects of various applications on science skills,
  - comparing effects of a specific kind of application (e.g., simulations) on lower level vs. higher level skills.

  A specially designed system architecture is needed to provide a scenario for comprehensively addressing and studying of the above simultaneously.

- A vastly unexplored area is computer applications in ESL.

English as a Second Language is one of the many issues that argues the case for individualized instructional programs. As an undergraduate the author worked with the Puerto Rican Family Association to help Spanish speaking students learn to read and write in Spanish, when they could not learn to read and write in English. Illiterate in their native tongue these children often times were considered retarded due to extremely low IQ scores.

A broad definition for instructional modulations must be made possible to account for cases where students may need to learn certain things in their native language first. While providing multi-lingual software to run under this or any other environment as part of selectiveness of educational tools, is by no means "a simple matter of programming", the possibility and feasibility are present.

- Word processing applications are fundamentally tied to instruction.

  Any educational system would contain its own desk-top publishing package, if no more than a keyboard and a printer. By incorporating the idea of levels of writing skills as part of modulation of functions of the word processing mechanism employed, even this fundamental facility can be improved upon.

- The parameters of assessing creativity and problem solving in instruction are described as:
  - comparing effects of computer vs. non-computer instructional programs on problem-solving and creativity,

- measuring the effects of applications on problem-solving abilities,
- evaluating the effects of various teaching methods and learning methods (e.g., discovery vs. structure),
- measuring the impact of instructional software specifically designed to improve problem-solving and creative skills,
- measuring the effects of usage on students with various characteristics.

  While the report examined focuses in this case on the use and effectiveness of Logo, the general aspects of the suggestive needed research covers those areas more broadly.

By incorporating all of the above into a single system the study of students' problem-solving strategies can be coordinated and compared.

- The Effects of computer use on attitudes and drop-out rates is another open area for investigation

  Computers can and have been known to help. However only if applied with massive impact on the student population will results be of significance. Therefore the system needed must be one that can be useful to many students for many purposes as fundamental requirement for massive applicability. The less theoretical aspect of massive applicability, reaching the classrooms, is a different issue relating to the same problem and one that is not ignored by this research. See Section 27, "Preliminary Implementation Suggestions" on page 185.

Needed Solutions

Thirty-eight reports and forty-four dissertations were studied in that report. One of the important findings deals with the wide variation of focuses, procedures, materials and reported results. The was point made.

> The absence of commonality makes it difficult to generalize across findings. The lack of standardization of data reported from study to study also severely hampers summarizing across studies.

Harmony, as an approach to educational computing has not been attained, perhaps has not been a goal of previous research by computer scientists. When computer-based instructional systems are narrowly focused and also market driven by current opportunities, they cannot possibly lend themselves to observations that can be logically correlated and standardized. The conclusions reached must then,

> ... be viewed tentative until more studies of similar types and with similar reporting styles are available to confirm these trends.

The problem succinctly stated is that the computer-based instructional systems available, and that can be studied, cannot provide similarities and standard relationships in terms of:

- content area,
- application type,
- grade level, and
- types of students.

A system is needed that can bring all the elements of needed research in this area together. Thus establishing the foundations to meet all of the requirements coherently.

Section 8. State-of-the-Art

What emerges from a survey of the field is two basic approaches to educational computing:

- Computer Managed Instruction (CMI),
- Computer Aided Instruction (CAI).

The Artificial Intelligence (AI) derivatives of those are:

- Intelligent Computer Managed Instruction (ICMI),
- Intelligent Computer Aided Instruction (ICAI).

Note: For a general discussion of AI see Section 2, "Necessary Information" on page 5.

Additionally combinations of both are found in single systems:

- CMI/CAI,
- ICMI/ICAI.

Current Approaches

More specifically:

- CMI - is a form of educational data processing.

Some goals of such systems are:

- interactive management mode,
- individualized program management,
- "automated" data collection,
- data processing,
- reporting.

CMI(s) emerged in the early 70's (Brudner, 72). These provide information and assessments that teachers can use to later adjust teaching techniques in the classroom. Ultimately the goal is to better help the students, by matching his needs with an educational program. But the students are not directly in contact with the process.

- CAI - is also a form of educational data processing.

Some goals of such systems are:

- interactive instructional mode,
- program delivery,
- data collection,
- data reporting.

Students are in actual contact with both the computer and the process. It could be argued that while this approach is more direct, it may be less influential overall, because it is limited by definition. This approach is one of the oldest. The term CAI already appeared in an early collection of articles by Haga (67).

- CMI/CAI - is a more comprehensive form of educational data processing.

  Combining the two paradigms seems to be the more natural approach to educational computing . It has been successfully done. Specificity appears to be the case when combining CMI and CAI due to the inherent complexity of bringing together conceptually many elements. Therefore examples of these are narrowly focused.

- ICMI - is an AI approach to CMI.

  In an effort to better handle the complex decision-making processes that inevitably appear, AI techniques have been incorporated into CMI software. (a form of decision making). Including such techniques as backtracking and rule firing explanation, these systems begin to address some of the more difficult aspects of the education management problem.

- ICAI - is an AI approach to CAI.

  There is more emphasis here on automatic adaptation of the software to the student, hence more usability. Solutions motivated may start with better representations for modeling the students. The few prototypes in this category e.g. SOPHIE, SCHOLAR, WUMPUS (*The Handbook of Artificial Intelligence*, Barr, 81) limit the use of instructional strategies.

- ICMI/ICAI - is a more comprehensive AI approach.

  The same benefits that derive from the blending of CMI with CAI are magnified with this approach. They combine the two important aspects of mod-

eling and adaptability. The goal, as it is in many cases where combinations of AI techniques are applied, is to enhance the adaptability of the system in this case attempting to more closely match the needs of the student. Such systems typically include representative student models. These, along with methods for isolating functionality, are attempts to automatically formulate what a student needs to learn, having decided in some fashion what the student already knows (Park, Perez, and Seidel, 87).

Again, since both CMI/CAI and ICMI/ICAI cases call for extensive research and testing, it is reasonable to expect only specific demonstrations of the approach. Such limit the scope, intensity and migration paths for the students. In short, these systems are designed in a way that only certain users, at certain times, receive the benefits of the approach. These are even more narrowly focused than conventional CMI/CAI attempts and even more scarce. One such effort by McGraw and Brown (88) designed to provide a narrow margin of instructional facilities and for a selected group, is an illustration.

This characteristic of knowledge-based systems has not been limited to educational knowledge-bases. It has been noted that MYCIN (Shortliffe, 76) for instance, operated on a narrow set of specialties (Barnett, 82).

Evolution of the Approaches

The representation that follows summarizes what can be viewed as an evolution of the more significant approaches. Each approach serves varieties of functions, captures specific needs, addresses relevant problems.

Ultimately this evolution reveals the definitions for a supportive structure that captures the essences of the previous attempts and extends them across educational programs. **A supportive structure capable of extending the power of all of the approaches and greatly generalizing their usefulness.**

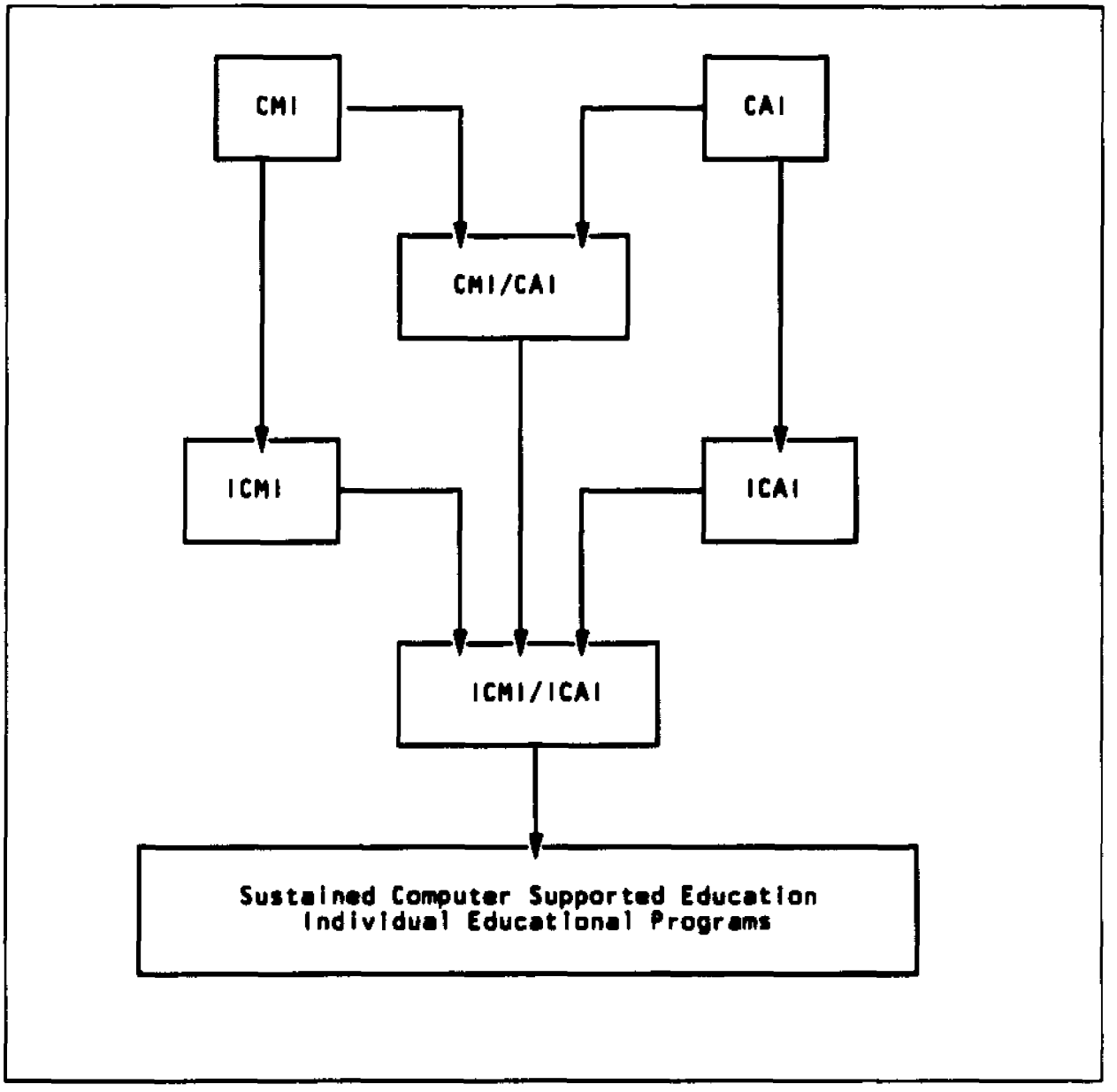

Figure 1. Various Approaches

The PLATO Approach

In a way the the PLATO Project (Olson and Magero, 76) had as its goal all of the elements that E^3 attempts to make feasible and useful. While similar in goal the two approaches are different in theory. The demonstration of a similar

approach under a new theory is thus an essential part of this research because, **the idea has not failed, but in fact only previous attempts to implement it have.**

PLATO, developed at the University of Illinois at Urbana, under the direction of Donald L. Bitzer, was designed to explore the possibilities of CAI in a host computing environment serving many remote workstations at different localities and providing a variety of educational services.

While PLATO was capable of adapting to many subject areas, it did not sufficiently provide a broad range of instructional adaptability. There were indications that the pace of the individual programs did not match the learning capabilities of the students using it. Plato did not adapt to the student. A formal evaluation on one PLATO course reported that students were evenly divided between those in favor of speeding up the program and those in favor of slowing it down, (Haga, 67). Clearly adaptability was a problem. But there were other problems.

The Problems with PLATO

The process of discovering and designing new computer applications is one of finding new things that people can do with computers. In industry this process has come to a sort of standstill and it is becoming increasingly difficult for software engineers to come up with new applications because most everything in business is already being done with computers.

The same is not true in the educational arena where a process which could be outlined, planned and designed for growth, just simply did not massively reach the classrooms. PLATO had the potential to evolve as a consistent process of mechanization of the classroom, very much in the fashion in which other applications in the industrial arenas were evolving.

It is consistently estimated that the following factors are the greater adversaries to the process of classroom automation as was exemplified in PLATO (Richter, 81):

- facilitation of the man-machine interface,
- cost efficiency, and
- user acceptance.

The implications in each case when not addressed, can prove catastrophic for the automation of the classroom. While in the industrial arena these things can and have been worked out, as the massive trend towards automation shows, in the educational area any one of these can still be singly responsible for, and become a complete show stopper. However at least one of the above problems must be re-evaluated for its validity.

The phrase "user acceptance" probably did not mean "end-user acceptance". User groups must be distinguished. In the case of PLATO, resistance by educators to CAI (Dick, 65; Silverman, 69; Splittgerber, 79; Watson, 72), means educational community acceptance. Students have not been known to reject the

concept of educational computing, but rather the educational community at large has (or had). Even early observers (Haga, 67) pointed the following interesting observation regarding at least one PLATO sub-project.

> Based upon follow-up interviews with members of the experimental group (end users), it was found that there were no instances where they would have preferred some other mode of teaching.

It was already clear then that students like computers as educational media, and there has been no reason for confusing the issues in that regard.

Examining the Problems

Let's consider the problems separately, before suggesting that an integrative solution is at hand.

Facilitation of the Man-Machine Interface

User friendliness or the lack there of, a relatively new concept, is a way of saying that computers are not meant for people. In trying to reach a production objective, accomplishing a certain goal with the use and help of the computer is a process of intensive effort. Only severe hackers and professional programmers can do useful things with computers. Computer-friendly applications are ways to reach the desire (same) goals within a larger number of extra steps. Exactly what happens is that what used to take one complicated and obscure step,

becomes ten (mostly complex and obscure) steps. The effectiveness of the approach is only symbolic in nature.

The many extra steps should in reality be taken from the beginning, that is, the system must be designed with the understanding that human beings will be using it, and must include in its design tools and techniques that facilitate such use. Allowing an educational application to evolve over time would not be equally effective. It would again limit its usefulness. **The idea is to provide flexibility that compensates for the many number of parameters that cannot be anticipated and also for the different kinds of users whose needs cannot be initially determined by the software designer.**

Profits that Surpass Cost

If any one area for computer application development is bound to fail for lack of funding, resources, research, it would be the educational area. The reason which is clearly funding is a great roadblock to say the least. The concept of host to workstation distribution of services already present in PLATO, is a valid one which has served industrial automation well. However it is the case that in that type of setting two things start to happen rather invitingly:

1. the number of serviced applications increases,

2. the complexity of each serviced application also increases.

The reason is again, that computer systems inherently evolve moving in the direction of doing more for more people. That is their reason for being, that is what makes them competitive, in a market dominated by competition.

While the multipurpose application approach embodied in PLATO was probably correct, its feasibility clearly declined with it, in spite of the fact that PLATO was designed to provide "educational applications without regard to cost". This issue (cost) which is recurrent in the literature has led many researchers to believe that the only real hope for educational computing rests in the use of Personal Computers (PCs) in the classroom. The entrance of the microcomputer into the field *supposedly* marking the revitalization of efforts towards the automation of the classroom, has steered researches and developers away from the centralized approach.

These issues tend to change however, as new systems are constantly developed that modify the definitions of cost and performance. **It is therefore more important to continue to concentrate on finding the correct solution, not the most currently feasible, or most currently fancy.** This may in the end be more profitable as well.

Community Acceptance

Community resistance was a battle fought and relatively won in the industrial setting where job security is a major issue, and recycling of the human element

involved was a long drawn out effort that continues to go on. As computer systems evolve, clerks, turned programmers now must become knowledge engineers or some other thing to survive the pace. It is logical to expect the same eras of confrontation to be repeated where educational computing is concerned, and with the same results.

But, it could be easily argued that no job is at stake here. The literature contains no references (so far) of abounding educational resources or excessive educational opportunities for our communities.

Reflecting on the Problems

Many studies conducted in the years that followed PLATO were devoted exclusively to the accurate assessment of the relative values of educational computing. These are difficult to put into practical perspective. But one undeniable fact is that students like computers and computer related activities. Our new generations are made up of creatures of great affinity to this kind of technology. The video games epidemic has proven that.

Unfortunately, there are many more considerations that affect the future of educational computing. There are those hopes founded on technologies that begin to seem more reachable; there are some real challenges to consider; and then there is also the question of how will industrial interests impact on this arena. These are profound questions, that require more detail study, but for the

purpose of this research at least some aspects of those questions must be explored.

AI Hopes

While there are many open questions which should not be dismissed, one that strikes educators and parents alike, is how to better employ this particular affinity young people have for electronic gadgets to better their lives and our national portfolio of technically qualified individuals. The tempting wish is to be able to leave the student alone with the computer, hoping that it will somehow guide the student in the discovery of knowledge. The other "wish" that comes tied in with that one is - can AI pull this off.

AI techniques have been effectively applied to other domains (e.g. medical diagnosis, oil drilling, process control), therefore hopes exist, but their promise must be examined realistically. In one paper on the application of expert systems technology to CAI (Hong, 89), the following were listed as components that an expert system "usually contain",

- inference engine,
- knowledge-base,
- knowledge representation and acquisition subsystem,
- working memory subsystem,
- natural language interface,

- explanation subsystem,
- debugging aids,
- input/output facilities,
- knowledge-based editors.

The question of whether or not an expert system can contain some or all of those features is meaningless to its effectiveness. The single most important aspect of an intelligent system is how well is understanding of the problem reflected in its design and approach.

Another paper entitled *Education and Computers*, by Schank and Slate (85), at least relates how to use AI in this area. The title of that paper is qualified with *An AI Perspective*, but except for a serendipitous reference to "reasoning by rule application" there is no AI jargon present. To make the point more clear, rule application refers in this case, to the way a child thinks. The focus is on the child, not on the use of the computer, or the programming paradigm.

Incorporating the achieved advances in computer technology that come directly and truly from an AI perspective is a function of finding out how children work best, not how computers do. Too much effort has been consistently put into examining how expert system paradigms work inside computer programs.

Included in that paper is the following front page and content outline from the New York Times (12/9/84): **School's Use of Computers Disappointing.**

- Computers are used largely to teach computer literacy...
- The biggest problem is lack of software for other subjects...
- Schools lack the means of identifying good software and training teachers to use it...
- Most computerized software is routine drill and practice...
- School software does not utilize the power of the machine, but instead mimics other media, such as books and projectors...

Those statements are to this day largely true, and they apply to educational computing in general, not just to that small subset of it implemented as intelligent systems. The point so far is well taken and vastly documented. **The answer is to make computers a vehicle for intellectual exploration and the students the explorers, since they alone have the energy, desire and courage to make it happen for themselves.**

Real Challenges

However the real challenge is not in providing inspiration and admonishment. The real challenge is to find ways to bring the technology to the most unlikely settings, where it is needed the most. This is not often pointed out. It is indisputably a hard job to deploy widespread use of computers for Education in

general. Therefore, it must be even more difficult to target the special needs population and the underprivileged populations. But nowhere are those advantages more badly needed and ways must be found to make it happen.

The Harford Courant recently reported (3/90) that during a gathering at Manchester Community College to discuss the future job market, and the education and job training that will be needed in the work force, it was disclosed that by the year 2000 the following will be the state of affairs.

- White men will make up fewer than 15% of new workers.
- Women, minority and immigrants will account for 82%.
- Future workers are now children living in poverty in the inner cities.
- The number of Americans between the ages of 15 and 29 will decrease by 11%, while the number of middle aged people will increase steadily, exacerbating the labor shortage.
- Corporations will have to deal with decreasing profit margins caused by labor shortage.

Those groups have to be brought up to speed educationally because they are our work force. Where our society is failing, technology has to lend a helping hand, to even the odds. This believe is commonly supported by many educators who feel that Education is a family affair and not a monopoly of the school systems.

And where families, specially in the inner cities, cannot sometimes provide the needed supportive educational structures, technology must do so.

Specific examples validate these feelings. In a report by Hotard and Cortez (88) the necessity of educational remediation for disadvantaged children is the case study. "Teaching groups of remedial children and addressing the needs of each child in the process is very difficult to achieve". But using a CAI in reading and math, remedial children, monitored through data collected over four years, "made gains above the national and state average in math. The gains in reading exceeded the average gains for the state and in most years the average national gains." Computers can help even the odds.

> The Computer Assisted Instruction Program has proven to be a consistently effective remedial tool.
>
> Remedial students acquire true proficiency and can solve many math or reading comprehension problems at their functioning levels.

A report of a different nature by Hofmeister and Ferrara (86) targeted as a problem the fact that:

> ... no literature or data existed which described the application of expert systems technology to special education.

Having addressed by means of specific and detailed prototypes both assessment and instruction in special education the report concludes as follows.

- There are important problems that could benefit from the application of expert systems technology.
- It is possible to develop practical exportable expert systems with:
  - tools presently available, and the
  - limited research and development resources available.

Industrial Interests

There are many current press references to industrial vested interest in education. Industrial arenas are concerned in two ways. Educational computing is potentially big business. It is estimated that education is a multibillion dollar a year market for information systems, with projected industry rates of $25 to $30 billion over the next 10 years, (IBM THINK Magazine, 88). While the national problem viewed as a business opportunity is appealing to industry, what is painful to corporations is the problem of finding qualified candidates to fill jobs now, also a derivative of the same situation. So interests are indeed paradoxically vested.

Taking advantage of the opportunities, several corporations have introduced software packages that rival in price and complexity with DP oriented software vendor packages. There is a (current top) licensing price tag of $53,200 for a CMI system by Goal Systems International, Inc., Columbus, Ohio, that even runs on an IBM 3090 under MVS. This type of computing scenario can be

made as an analog to environments used by large corporations for payroll or billing, for instance. In other words heavy-duty computing. The company has a 24-hour, toll free number and a technical support department. (T.H.E. JOURNAL, Dec./Jan. 88/89).

There are other software packages listed in the above cited article, all are CMI(s). These systems (CMI) offer the greater promise, over a larger set of customers for substantial profit. They can extend over a school district aiding with some of the more empirical educational labors. Where public and legislative pressure has increased to demand accountability by the schools for quality education, systems such as these begin to lift some of the burden from the educational team, providing ways to focus on the real problem, the students.

The fact remains that those best suited for using computers in the educational setting, that is the students, still get little if any of it. CAI(s) which more directly put the computers in the students' hands, are not always offered in such packages. It is understood that commercially provided CMI(s) will continue to find their logical connections to CAI(s). But no system built yet has planned for this in its conception comprehensively, due in part to commercial factors. Some of these systems must get out the door, make money, then evolve.

Drawing Some Conclusions

The introduction of computers into the classroom, as exemplified by PLATO, and less significantly by other more ad-hoc attempts, have failed, because they have failed to be massively adopted. Today it is unthinkable, except by the very primitive, to employ any other means but spreadsheets and word processing to accomplish the tasks performed by those computer-based tools. However there is pervasive use of paper and pencil in the classroom, while at the same time there is a shortage of teachers and unnecessary run-away mediocrity among a generation of young people who belong to the technology.

In the world of Education, evolution of computer use is an unlikely path towards massive implementation. It cannot be emphasized enough that educators have been in business for many more years than computers have. What the technology is capable of doing, and how this would benefit Education is already common knowledge. The educational community knows what the technology should do. What needs to be discovered is how to best use the technology to realize those expectations. **An approach that will reach the classrooms massively must be potentially capable from the onset to meet all needs.**

Section 9. One Formal Answer

While federal law makers still debate the definitions and legal rights of students with special needs, some State Laws have provisions that attempt to deal with adequate individual education for students.

> Now that we have achieved education for all,
> let us seek education for each ... (Pfeiffer, 68)

Unfortunately the Law requires this only when deemed necessary. In the State of New York and as contained in Part 200 of the Regulations of the Commissioner of Education, a Committee on Special Education (CSE) is formed and proceeds accordingly, via established guidelines, for any (and conceptually each) student when special needs are evident. (*A Guidebook for Committees on Special Education in New York State*, 1987).

The CSE Process

CSE is a term broadly used to relate procedures, people and laws brought together in an effort to protect students' right to quality education. The legal aspects of this process are wide and complex. *McKinney's Consolidated Laws of New York, Book 16*, contains many articles concerning specific forms of handicapping and learning disabilities. These are ways of enforcing "problem solving" approaches which is equitable and formalized.

The consequences of having formalized an approach to special education under the law is not without implications to Computer Science. It is important to consider or at least speculate, that having in fact formalized the process under the law, and for the explicit purpose of protecting the individual rights of the student, the possibility of transferring such processes to a mechanical device, such as a computer, becomes an unlikely task, a task no more possible perhaps, than it would be to reduce a jury of twelve peers to a piece of software.

It is not suggested here that a computer program, expert or not may or could substitute the human factors involved in the creation of a CSE, but rather take inspiration in the activities and roles of the various CSE members, to help improve on the imperfections of computer-based education, as functions of both Computer Managed Instruction and Computer Aided Instruction.

Definitions, Roles and Descriptions

In this section the term CSE will be used as in Committee on Special Education as it is known and understood by the educational community, in the State of New York.

The CSE is an interdisciplinary team that may recommend an appropriate Individualized Educational Program (IEP) based on evaluations. An IEP is a formal attempt by the educational community, at least under the Educational Laws of New York State, to meet the individual educational needs of a student

as decided by the CSE. The program is implemented upon parental approval. The CSE is further involved in planning conferences with the parent upon start of the IEP and upon annual review. A triennial evaluation is provided that determines continued eligibility for the program, and also current appropriateness of the program. Initial referral to the CSE is based upon suspected handicapping conditions affecting the student in some way. The following list outlines the typical make-up and activities of a CSE.

Team Members work in the spirit of cooperation, to arrive at decisions that have great impact on the educational welfare of the student.

Specialists bring special skills to the process.

Leaders protect the interest of the student.

Problem Solvers suggest solutions.

Decision Makers determine eligibility and appropriateness of programs.

Program Analyzers review the adequacy of special programs, program descriptions and student data.

Short-term Planners conduct evaluations and make recommendations within a specific time frame, affecting the immediate future of the student.

Long-term Planners plan ahead for student opportunities, alternative programs or services, to help the student achieve long-term goals.

Resource Persons assist school staff, parents and students involved in the process.

Mandated Members are the following:

- the school psychologist,
- a teacher or administrator of special education,
- the school physician (if requested),
- the parent(s) of the student.

Article 89 of Educational Law specifies that the above be appointed to the CSE. The participation of the student's teacher is also mandated.

CSE Management Strategies

Officially suggested strategies are commonly put into practice to help CSE members implement their responsibilities. Some are as general as pointers to deal with the committee itself. Others are more specific and refer to support from clerical staffs and computer storage information, presumably when available.

Other strategies refer more exactly to computer related services such as those provided by the Board of Cooperative Educational Services (BOCES). BOCES

provides and maintains computer records and evaluations that guide the design and development of an IEP. A document produced by BOCES known as the *The Living Document* is used throughout the entire CSE/IEP process, which is at least a three year cycle. It is suggested that BOCES representatives participate in the CSE process as well. It is further suggested that activities and practices of other CSEs in other districts be monitored and communicated with.

Appropriately insuring that the IEP(s) are carried out with the participation of the student's teacher is strategic to the process. Providing student instruction and evaluation, 12-month programming, continued education, are all specifically outlined as directives. Even more interesting is the fact that innovative school practices is considered a strategy in the process.

These sensible procedures nevertheless spell out a well organized logical set of routine activities that put to motion in the school setting, ultimately provide special education where special needs exist. An educational plan for the child is in fact produced by means of impartial due process.

Stages of Implementation

Management strategies during the CSE process are implemented at various stages and accordingly. But, the fundamental steps in the CSE process and their implications are as follows.

Referral is done via parent concerns, teacher observations, or more obvious indications of a situation that calls for more careful analysis.

Evaluation is typically compiled from various forms of scholastic or psychological testing.

Recommendation is a formal plan for remediation, or dismissal from the process for the current time.

Implementation is an attempt to carry out out the remediation plan.

Planning Conferences,

Annual Reviews, and

Triennial Review are strategies for assessment.

Structure of the CSE Process

The process was studied for the purpose of drawing upon its steps and strategies as foundation for a computer system with related purposes. At each step (listed above) highly specialized skills and/or issues play vital roles. For the purpose of this research the structure of the process is of greater importance overall than the details of it. Informally for now, evaluation, recommendation and implementation are in themselves a form of information flow that lends itself well to

computer implementation, while the others do not (so easily) and require further examination. It will be seen later that E^3 attempts to incorporate precisely those kinds of information flows directly, while simulating the others or less directly carrying them out.

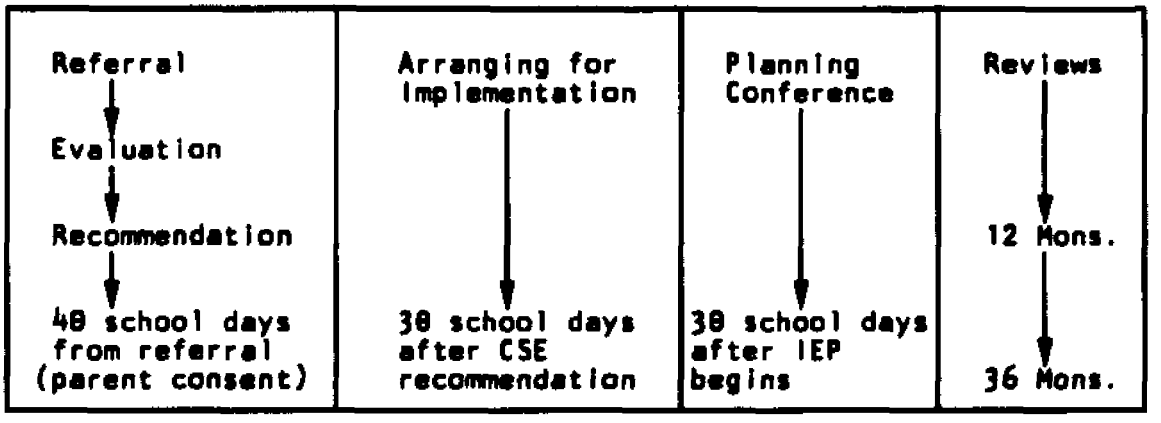

Figure 2. The CSE Process

According to New York Educational Law the process as outlined above carries time frame constraints, which are imposed, (recall) in order to ensure that the student's educational needs will be provided for. The goal of all this is to match a student's individual needs with an educational program expediently.

Concerns With the Process

While in New York State standards for CSE are kept high, in general these types of processes are not without fault.

The delays that accompany any form of consensus are bound to affect the effectiveness of the application. Interruptions in the student's educational program in general have been noticed to accompany the hearing process (Parry *et al.*, 85). Scheduling the various aspects of the process to limit, if not avoid delay, can be difficult (Smith, 81). While mandated, these processes are nevertheless subject to the human element involved.

Another concern is always cost. At least one cost analysis reviewed (Henderson and Hage, 79) showed the average cost (at the time the report was published) to range from $2000 to $4000. Thus, Some estimates indicate that it may cost twice as much to educate a special needs student than an average student (Belsches-Simmons and Lines, 84).

In addition, it has been debated whether or not impartiality is attained (Salend & Zirkel, 84). The background of hearing officers (in general, but not necessarily in New York State) vary greatly. One hearing reportedly included a homemaker, postmaster and research biologist (Turnbull, Strickland and Turnbull, 81).

While again, some states are more on top of these mandates than others, even those great variations among states have been identified as a problem (Smith, 81). To summarize the situation, current processes do not necessarily ensure equitable and effective educational decisions (Salend and Zirkel, 84).

Section 10. The E^3 Approach

The CSE Process of the educational setting along with a combined ICMI/ICAI approach are the foundations for the design of E^3. By mapping the strategies employed in the CSE Process, as it takes place in the educational setting, into blocks of knowledge (represented in some form such as *in rules*) and by grouping and invoking these blocks functionally, the processes can be approximated with software under an architecture. It is not suggested that this is computer-based special education, but merely that special education with its focus on individualized instruction provides inspiration and sets the goals for this approach to computer-based education in general.

Mappings of Related Tasks

The tasks related to the CSE process as they take place in the educational setting were outlined earlier, as were the goals implicit in many CMI and CAI systems. This information can now be used to map the human tasks to the closest computer systems approximation, interpreted as implicit functions of these.

Referral as a form of services invocation is implicit in the use of either CMI or CAI approach.

Evaluation as a form of diagnosis, planning or assessment is related to CMI(s) services.

Recommendation as a form of curriculum planning is also related to CMI(s) services.

Implementation or carrying out the plan can be made as an analog of CAI(s) application.

Planning Conferences as a form of instructional management is related to CMI(s) services.

Annual Reviews also forms of instructional management are related to CMI(s) services as well.

Triennial Review another form of instructional management is also related to services provided by CMI(s).

For a truly integrative system all of these tasks must be implicit in the design architecture.

Computer Supported Education (CSE) is now re-introduced as a term that describes what a combination of CMI and (Adaptive)CAI software could be made to accomplish within an integrative and organizing structure.

The functions of both the CMI and CAI paradigms can be combined by means of what could be labelled a psycho-educational knowledge-base. The automatic production of IEPs then becomes a product of the structure as well as a the primary goal of the system. But before proceeding with the description of the E^3 architecture, it may be useful to discuss some of the issues that were taken into consideration and that relate to the design decision.

Fundamentals of the Approach

A knowledge-base approach is valid for this application. The heuristic nature of the processes appears suitable for representation with AI heuristic-based tools. Another reason is the fact that computer-based educational systems have already been escalated to the level of "intelligent" systems as seen earlier. It would be unwise to ignore that trend, especially because their applicability has been demonstrated by research (McGraw and Brown, 88). Section 2, "Necessary Information" on page 5 offers a brief overview on how Knowledge-based Systems typically work.

Knowledge-based approaches seem naturally suited to capture the kinds of information and heuristic processes that must interact here. These systems use explicitly represented knowledge (attempting) to obtain levels of decision making performance in some ways similar to those of a human expert. Some examples of such systems are MYCIN which diagnoses blood infection, Dendral which analyzes mass spectographs, Prospector which aids in oil drilling (Barr and

Feigenbaum, 81) and YES/MVS an experimental expert system that assists with the operations of a large computer complex in realtime (Griesmer *et al.*, 84 and Milliken *et al.*, 86). These are general examples in the literature But, there are references that share issues with the E^3 application.

It has been demonstrated (Cruise *et al.*, 86) that complicated programming tasks can be broken into isolated modular units called *productions* or *rules* which express actions that are taken under certain conditions. A separate program called the *inference engine* checks the conditions associated with each rule and executes the corresponding action, when the condition is satisfied.

Because it is often difficult to express even a single granule of knowledge in one rule, some knowledge-base tools support hierarchical structuring of rule blocks. These language constructs facilitate the organization of a large and complex knowledge-base. In general, a functionally partitioned knowledge-base approach implemented with rules is appropriate to the complexity of this application.

More Specifics of the Approach

The architecture of E^3 can take full advantage of the approach discussed above and of constructs in computer languages that supports them. In Cruise *et al.* (86) a block oriented language is introduced that supports both rule blocks and

procedural blocks. Such an integrated approach becomes very useful, almost ideal for this application.

Rule blocks are rule-based programming structures useful in relating similar groups of knowledge in a hierarchical organization. In this way the knowledge-base can be organized logically.

Tasks of related knowledge can be organized into one rule block. Some of the knowledge used in the psychological assessment of students, for instance, is related and can be organized into one such block. Similarly other forms of assessment can be captured in this manner. Inferencing mechanisms are applied separately to each rule block, creating a systematic decision making machinery.

The other type of blocks, procedural blocks, are extremely useful, and particularly so for this type of application, since they can embody large amounts of conventional procedural software of the kind normally used in the implementation of instructional software. These procedural blocks are activated as actions which are specified in the condition/action set. See Section 2, "Necessary Information" on page 5 for more detail.

More specifically the idea is to take advantage of the fact that CSE processes have already been broken down into functional components by the educational community. It is possible to capture the essence of those kinds of functions with

structured blocks of knowledge. This design favors the development of instructional software independently by programmers and educators.

However it must stressed that it is not in the particular programming paradigm chosen, but in dissecting and understanding the problem that the offered solution is based upon.

In Search of a Formal Model

There is no structure that can be formally applied to define the needs of a student and match those to educational software. Conceptually such structure should be:

- formal,
- analytical, and
- studied over time.

It would be defined in terms of:

- learners,
- needs of the learners, and
- instructional materials.

Such a model would include all variables that operate on those terms. Which are at least:

- the set of learners,
- the characteristics of the instructional needs for each learner,
- the set of applicable instructional materials for each learner's characteristic needs,
- the effect that the passage of time has on the needs of each learner, given the application of instructional materials.

Studying the validity of such formalism over time would then be necessary for:

- determining the needs of any learner at any given time, based on the current frame of reference for that learner,
- specifying exactly how useful any given instructional is at any given time for a given learner, based on the current set of goals for that learner.

With such a model it would be possible to actually characterize the learners and the instructional materials, and be able to match them more precisely. That is, it would be possible to formally produce an exact match.

This kind of formalism is speculative, in that educational processes do not lend themselves to analytical tractability, and may in fact be intractable. This does not impede the design of a system upon the premise that this formalism can be approximated and is needed to derive more precise usability from the application of computers to educational processes. **The alternative approximation would be heuristic in nature. The E^3 model is such an alternative.**

Part Three: E^3 Architecture

Section 11. Functional Organization

This system is organized (at a high level) by functions. It consists of three functional platforms for the development of software. These platforms correspond to the elements in the individual education process, grouped into:

- information about the pieces that make-up the process,
- knowledge on how to use that information, and
- curricula.

Because these functions are complex, each platform is composed of one or more component and each component is made-up of an unspecified number of sub-components. Enhancing E^3 involves adding components and sub-components when necessary. The system is therefore structurally designed for flexibility, and openendedness.

This kind of functional organization has been proven useful in the design of knowledge-based system (Ennis *et al.*, 86), and is extended here more formally to the procedural elements of the system in a consistent design approach.

Functional Platforms

The functions of the system are organized in three platforms:

The Computer Supported Education Platform (CSE), which contains the knowledge-base,

The Adaptive Computer Assisted Instruction Platform (ACAI) which contains curricula, and

The Livingbase Platform (LB) which contains the base of data.

The CSE Platform

The **Computer Supported Education (CSE) Platform** is in itself a Psycho-educational Knowledge-based System. It is organized into six components, or more specifically, *sub-domains*. In *Production Systems* grouping of knowledge is a technique used to establish effort foci to help deal with the difficulties that emulating a real life activity poses. This style is a natural for E^3. There are other examples of this technique, in however unrelated but also highly subjective domains, such as was done in HEARSAY-II (*Artificial Intelligence*, Winston, 79). In E^3 the domains are loosely related to activities that typically take place in the educational environment, when a Committee for Special Education is formed in order to assess and plan a program that meets the student's needs.

Hoping to capture and make the benefits of highly specialized knowledge resources more widely available, those activities are accounted for in the design and grouped together as described in the next section.

CSE Components

The Controller as its name implies, is used to control the system, not just the CSE, but the entire E^3. It then exercises control over its operational environment. It invokes the functions of all the other components. It alone effects the interface among the platforms and between itself and the other components of the CSE. It alone interfaces directly with the student, the educator and the technical support person.

The Controller captures more of the influence of the computer technology employed. While the other components attempt in some ways to represent activities carried out by various member of an educational staff, the controller is an autonomous addition to the educational process. This is generally sound in the development of large complex system, but in particular for knowledge-based systems it has been suggested before to make the process control functions a separate and active domain (Barnett, 82).

The Counsellor concentrates on the student, his strengths, interests, weaknesses and needs. It must examine all information relating to the student and help

decide the best possible initial path to follow. This means that the Counsellor coordinates the up-front decisions that guide the students during their interactions with E^3, and with the selection of currently appropriate educational goals.

The Evaluator gets involved with the selection of instructional tools. It must know what these can and cannot accomplish and how. Since this is done after an initial path has been formulated by the student in collaboration with the Counsellor, the Evaluator will only concentrate in finding an instructional tool that meets the demands of the currently decided upon educational goals.

The Evaluator examines information on what is available in terms of adaptive curricula and selects them appropriately to meet the student's needs.

The Specialist selects a set of appropriate instructional strategies to compliment the decisions made by the Evaluator

The Specialist concentrates on the decisions that have so far been made. Namely based on the current educational goals and working with the selected instructional tools, it tries to modulate the session by employing the most appropriate available instructional strategy that can accompany the selected instructional.

The Frameworker is the vehicle through which the individually tailored learning experiences are created. The Frameworker receives all the information it needs

to backup the session with appropriate mechanisms and it works exclusively on generating and presenting the session.

The Monitor keeps tracks of progress and extracts information that may be useful for better understanding the student and for sustained assessment of the instructional tools being employed.

Optional CSE Components

It could be argued that two more pieces of logic are needed to effectively complete the overall picture of a totally comprehensive educational environment. These are labelled "optional" because they do not enhance the activity of preparing IEPs, but they provide other kinds of important functions as will be seen.

These activities are implicit parts of the processes as they are carried out by human beings in the educational setting. As described below, they do take place and are part of the processes, therefore counterpart functions should be conceptually present.

The Coordinator is an assessment tool. When a particular piece of instructional software, or instructional modulation is applied, its effectiveness (based on criteria associated with the instructional) against the given goals is detected by the Monitor, and stored in the LB. Analysis of this information by another

knowledge-base component is effectively a form of automatic evaluation of instructional software.

Information about what seems to work in each particular case (or not work) is always collected as part of the purpose of this system, for more detail (off-line) studies of an empirical nature. These are the more precise forms of assessment and effectiveness evaluations that E^3 means to support. However, an immediate (on-line) coordination facility could greatly enhance the usefulness and re-application of the instructional software.

The Advisor is a vehicle though which recommendations can be sought from the system by students, parents and in particular educators. This component assumes that one of these things or both is true:

- the information sought (advice) is contained in the hosting E^3, but

  if the hosting E^3 cannot within its knowledge provide sufficient advice, then

- there is another E^3 somewhere that may be able to provide counsel.

In essence the knowledge and sophistication of any given hosting E^3 depends on what knowledge has been amassed in its knowledge-base, and this knowledge would vary from system to system and according to applied educational preferences. Other hosting E^3 Systems could be better equipped to handle specific educational situations, and their knowledge should be available for consultation.

The Oracle is invoked by the Advisor to obtain knowledge and information from another E^3. This is basically a communications device that provides interfaces with other known E^3 systems. Ideally this component (alone or in collaboration with the Advisor) should contain components capable of:

- communicating with other hosting E^3 systems (sending/receiving information),
- accessing information by triggering another E^3 as a remote end user,
- testing and evaluating users remotely,
- producing IEP(s) remotely,
- acquiring additional software.

It is beyond the scope of this research to elaborate on the details of these activities, but it is important to recognize that they are viable functions that can conceptually be integrated.

The ACAI Platform

The **Adaptive Computer Assisted Instruction (ACAI) Platform** contains instructional software in decomposed form. Embodied in the knowledge-base of the hosting E^3, the style and intensity of delivery are part of the instructional subcomponents in the ACAI. Therefore specific instructional and modulation subcomponents are selected by the educational staff (from what is available, which is conceptually open-ended). The goal is then not only personalized instruction,

but also tailored management. Each hosting E^3 can be made as a reflection of what the educational staff believes in and uses in the classroom.

In this way grouping and combining instructional software by function, makes the total value of the approach greater than the sum of its parts. In this case integration of CMI with CAI is aided by the separation of the *instructional modulations* from the actual *instructional agenda.*

The ACAI Platform is then meant to produce programs which are in some ways analogous to the individual educational programs formulated and used in the educational setting, and superior to conventional CAI. The ACAI is in fact two separate sets of software components.

ACAI Components

The Instructionals component is made-up of software tools developed to target specific instructional goals and sub-goals, to provide various subjects and levels of sophistication within those subjects (topics and subtopics).

The Modulations components contains sets of instructional strategies developed to compliment instructional tools with specific learning strategies, further focusing on the individuality of the student. When one Instructional sub-component and one Modulation sub-component are assembled together they become a single and unique ACAI temporarily existing component. This can be

a very simple and straight forward combination, but it can also take forms that provide detailed degrees of adaptability.

The LB Platform

The **Livingbase (LB) Platform** is a continuously changing storage of information, designed to support the modeling scheme imbedded in the knowledge-base. It is this information that the knowledge-base components use to formulate decisions, and to create and re-create learner models.

Designed and named in the spirit of the BOCES Living Document, the name in both cases seems appropriately chosen because the LB is constantly undergoing change, as the student is, and in response to the learning processes.

The LB is made up of largely expandable components used to maintain readily available and up-to-date information on students, educational goals and instructional tools. It is very active during use and refinement of the system. Its organization of the LB is vitally important to the system. It is also broken down into components.

LB Components

The Learners or Learner-base, contains learner profiles and educational goals and sub-goals specified for the students. It includes his needs, strengths, interests, and in general can contain insight as well as routine information on the learner. This information can be initially predefined by either an evaluation made by the educational staff or gathered from test scores maintained by the educational staff. It can be made of of standard assessment test results or more specific forms of assessment. It can be as general as a report card if no other data becomes available. However, as E^3 becomes involved in the educational process for each particular learner, both the profile and the educational goals and sub-goals are refined.

The information effectively becomes a road map of education for each learner in particular. Each profile is maintained individually and in response to ongoing interactions promoting educational experiences for each learner that become uniquely different.

The Instructionals or Instructional-base contains profiles on the existing instructional tools of the ACAI. Specifically it is concerned with the instructional part of the ACAI available to the current E^3. It includes information that enables E^3 to match the current educational goals and sub-goals for the learner with the projected instructional goals and sub-goals that can be attained via specific instructional sub-components.

The projected ability of the tools to meet those goals is also constantly being updated. Thus usage and applicability can be reassessed to determine relative effectiveness. In this way it is capable of altering the projections in such a way that unanticipated usefulness of the tools may be discovered and lack of effectiveness taken into consideration. Potentially the discriminating capacity of the system increases with use and the sophistication of the data recorded here.

The Modulations or Modulation-base contains profiles on the existing instructional modulations for the instructional tools of the ACAI, in addition to the information contained in the Instructional-base. Specifically it is concerned with the modulation part of the ACAI available to the current E^3. It includes information that makes it possible to enhance instructional tools with a variety of delivery and presentation strategies (based on sound educational practices) and prepared to go along with specific instructional tools.

This component is also concerned with the projected use and effectiveness of the instructional modulations of the ACAI. It works like the Instructionals and it is updated and/or modified in the same fashion and for the same reasons.

The Inventory is an inventory of instructional and modulation tools that are present in the system. Assessments of the relative effectiveness rating of the educational/instructional software could also be recorded in this component.

The Inventory is also meant as a control mechanism (or audit trail) to recognize software, versions and other relevant information. This facility helps with the explicitly administrative functions associated with educational processes.

Section 12. Software Organization

The software organization of the system is parallel to its functional organization. Each functional platform represents a *software layer* or given set of components. It convenient to view the ACAI Platform as existing in two separate layers of software.

Software Layers

The software organization of the system is defined as follows.

The Computer Supported Education Layer contains the knowledgebase components of the CSE Platform.

The Instructional Layer contains the instructional sub-components of the ACAI Platform.

The Modulations Layer contains the modulations sub-components of the ACAI Platform.

The Livingbase Layer contains the database components of in the LB Platform.

Layers are logical groupings of components for the purpose of software development.

The CSE Software Layer

The CSE Layer contains software to perform all of the functions of the CSE Platform. It can be designed and implemented as one module divided into blocks corresponding to each of the CSE components, or it can be designed as separate but collaborating modules.

The Instructional Software Layer

The Instructional Layer contains any amount, as selected by the current educational staff, of instructional software. This software must be *originally designed, or adapted* to operate in this environment.

The Modulation Software Layer

The Modulation Layer contains corresponding instructional strategies that guide usage of the instructional software. That is, information and facilities to modulate the respective instructional in terms of intensity, quantity, teaching strategy and other applicable criteria. This is again done in collaboration with the Instructionals.

The Livingbase Software Layer

The Livingbase Layer is the main storage of information.

What the Organization Represents

The figure that follows is a conceptual representation of the organization of the system. It also discretely spells out pieces of software necessary to make up the system.

Each software layer is made up of software components to carry out the functions required by the organization (or functional platforms of the system). The CSE layer, for example contains the six software components necessary to implement the functions of the CSE. Likewise the LB layer contains distinct pieces of software, which correspond to the four databases.

The ACAI layer, however is made-up of *virtual components*. That is, while the components exists they do not exist in useful form, but are made useful (put together or created) in realtime when needed. For instance,

$$\{ACAI(A_1) <= I(A_1) \ \& \ M(A_1)\}.$$

In this example a *currently existing* ACAI component (for Arithmetic), was drawn out (created) from a single Instructional sub-component $I(A_1)$ and a single modulation sub-component $M(A_1)$.

More specifically, the ACAI Layer requires two distinct pieces (or sets of software):

- instructional sub-components, and
- modulation sub-components.

The Instructional Layer can contain instructional software much in the flavor of conventional CAI(s). The Modulations are strategies and facilities often also contained in conventional CAI(s). The difference here is that the associations are made according to individual student needs evaluated in realtime.

An instructional sub-component, therefore, is not applied by itself but must be combined with a modulation sub-component before delivery. The decision on how to arrive at a suitable combination is made by the CSE. For this purpose it is necessary that each instructional has a set of modulations associated with it.

For example,

$I(A_1)$ meaning Instructional in Arithmetic 1

can be applied in combination with its own set of strategies which are contained in the Modulation Layer sub-component:

$M(A_1)$ or Modulation for Arithmetic 1.

The application of these results in the following ACAI component:

$\{ACAI(A_1) <= I(A_1) \ \& \ M(A_1)\}$.

In the figure that follows, each block $I(A_1)$, $I(A_2)$, etc. is an instructional topic. So that A_1 would be distinctly different, or different enough in some way to be in a separate block of instructional software. Corresponding modulations are represented in the same manner and are associated with their respective I(s).

It is by combining these two sub-components that a single ACAI component exists. This is why ACAI components are referred to as virtual. Other subject areas and/or topics are signified by the letter X in the illustration.

CSE Platform

- Controller
- Counsellor
- Evaluator
- Specialist
- Frameworker
- Monitor

ACAI Platform

Instructional

$I(A_1)$	$I(A_2)$	$I(A_3)$	.....	.....	$I(A_i)$
$I(X_1)$	$I(X_2)$	$I(X_3)$	.....	.....	$I(X_i)$

Modulation

$M(A_1)$	$M(A_2)$	$M(A_3)$	.....	.....	$M(A_j)$
$M(X_1)$	$M(X_2)$	$M(X_3)$	.....	.....	$M(X_j)$

LivingBase Platform

- Learners
- Instructionals
- Modulations
- Inventory

Figure 3. The E³ Organization

Section 13. Activities by Component

The following list gives an idea of some of the types of things that the various E^3 Components should do. These are only inspirational and must be subjected to more specific design criteria at implementation time.

Activities of the CSE Components

- The Controller
  - controls all interactions within the system, and
  - provides peripheral resources.

- The Counsellor
  - collaborates with the student in the decision process,
  - guides learning objectives, and
  - attempts to match the student with learning goals.

- The Evaluator
  - conducts an instructional analysis,
  - navigates instructional possibilities, and
  - arrives at instructional sequences.

- The Specialist
  - understands the needs and orientation of the learner,

- assesses learning rates,
- provides varying amounts of motivational strategies, and
- develops an instructional strategy.

- The Frameworker
  - individualizes presentations, and
  - supplements the instructional resources.

- The Monitor
  - monitors learner performance,
  - monitors E^3 performance,
  - extracts data from the process,
  - records diagnostically useful facts,
  - supports accurate recording, and
  - provides many forms of reports.

Activities of the Livingbase Components

These activities are loosely related to records kept in the LB. These are possible data records illustrative of the information that would be maintained.

- The Learner-base contains
  - learner information,
  - observed mastery evaluations,
  - generated mastery evaluations, and
  - mastery evaluation values.

- The Instructional-base contains
  - instructional component information,
  - estimated effectiveness evaluations,
  - generated effectiveness evaluations, and
  - effectiveness evaluation values.

- the Modulation-base contains
  - modulation sub-component information,
  - anticipated usage evaluations,
  - generated usage evaluations, and
  - effectiveness evaluation values.

- The Inventory-base contains
  - existing educational/instructional goals,
  - existing educational/instructional sub-goals, and
  - generated educational/instructional requirements.

By extrapolating these types of information, operational software may be made to produce unexpected results. For example *anticipated evaluations* may refer to those things which the particular software has as its goals. But *generated evaluations* reflect those things which the software has been known to accomplish in practice. In some cases even the developers of the specific instructionals may not be able to anticipate practical results or fully comprehend the current func-

tions the software provides. With this type of closely monitored usage more accurate assessment becomes possible. The scheme is also a discriminating mechanism that can be used to test or try out instructional software in general.

Activities of the ACAI Instructional Sub-components

Some examples are given below, however these sub-components are mainly a vehicle for the exploration of Computer Supported Education.

- Instructional Situations
  - simulations
  - discovery laboratory
  - problem solving
  - drills and practice
  - tutorials
  - demonstrations
  - games
  - etc.

These are situational contexts where particular instructionals can exist. Instructionals would be focused, possibly organized by topic within a given situational context. Particular instructional contents would cover the educational disciplines (Math, Science, Language, etc.) for example.

Activities of the ACAI Modulation Sub-components

Some examples are given below, but clearly the presentation media and current technology will determine the course of instructional modulations.

- Modulation strategies could be:
  - teaching strategies,
  - sensory strategies,
  - foreign languages resources,
  - emotional adaptability, etc.

Detail implementation of strategies could correspond to such things as reordering of phonemes, enlarging letter sizes, providing graphic flavored Math problems, special effects, multimedia presentation services, etc. when possible and applicable.

Part Four: E³ Methodology

Section 14. CSE Process Scheme

There are special processing schemes associated with the CSE.

The following overall process flow description is related to the representation that follows it and illustrates what the system does.

E^3 Process Flow

- (1) The learner enters E^3.
- (2) The Controller takes the following actions:
  - engages the learner, and
  - activates other CSE components.
- (3) the Counsellor takes the following actions:
  - interacts with the Learner-base,
  - selects relevant records,
  - maps learner information to instructional goals, and
  - produces a suitable *set* of educational considerations that are currently appropriate for the learner.
- (4) The Evaluator takes the following actions:
  - interacts with the Instructional-base,

- selects relevant records,
- maps instructional tools information to instructional goals, and
- chooses a suitable *set* of instructionals.

- (5) The Specialist takes the following actions:
  - interacts with the Modulation-base,
  - selects relevant records,
  - maps instructionals to instructional modulation strategies.

- (6a) A *Consideration Set* is created.

  (6b) A set of possibilities made up of considerations, instructionals and modulations should be available at this point, not specific ones. This is necessary for establishing current relevancy that can be monitored. If a single possibility was to be arrived at then there would be no need to monitor, just simply to report results. Best fit educational situations should not in this environment be *assigned* but rather *probed.*

  The purpose of a Consideration Set is discussed later in this section. To clarify references to it at the moment, it is sufficient to state, in AI terms, that a Consideration Set is equivalent to a *conflict set*, but used more carefully.

- (7) The Frameworker takes the following actions:
  - examines the Consideration Set which contains:
    - instructional considerations (from Counsellor actions),

  - instructional goals (from Evaluator actions), and
  - instructional modulations (from Specialist actions).

- Then alone or in collaboration with several modules invoked as actions, prepares or frames one of the possible educational sessions. This is viewed as a form of *tentative resolution*. Other possibilities are kept in the Consideration Set and may be applied subsequently.

- (8) The Monitor takes the following actions:

  - accepts data during the session,
  - generates LB records and temporarily stores them;
  - it may then decide to re-invoke the Frameworker causing a different tentative resolution to be chosen,
  - or it may re-trigger the Controller to re-create a new Consideration Set, that is, to arrive at a new set of considerations, instructionals and modulations.
  - This effectively creates a shifting sets environment.

- (1) The learner disengages E^3.

- (2) The Controller takes the following actions:

  - disengages the learner,
  - causes an update of LB records.

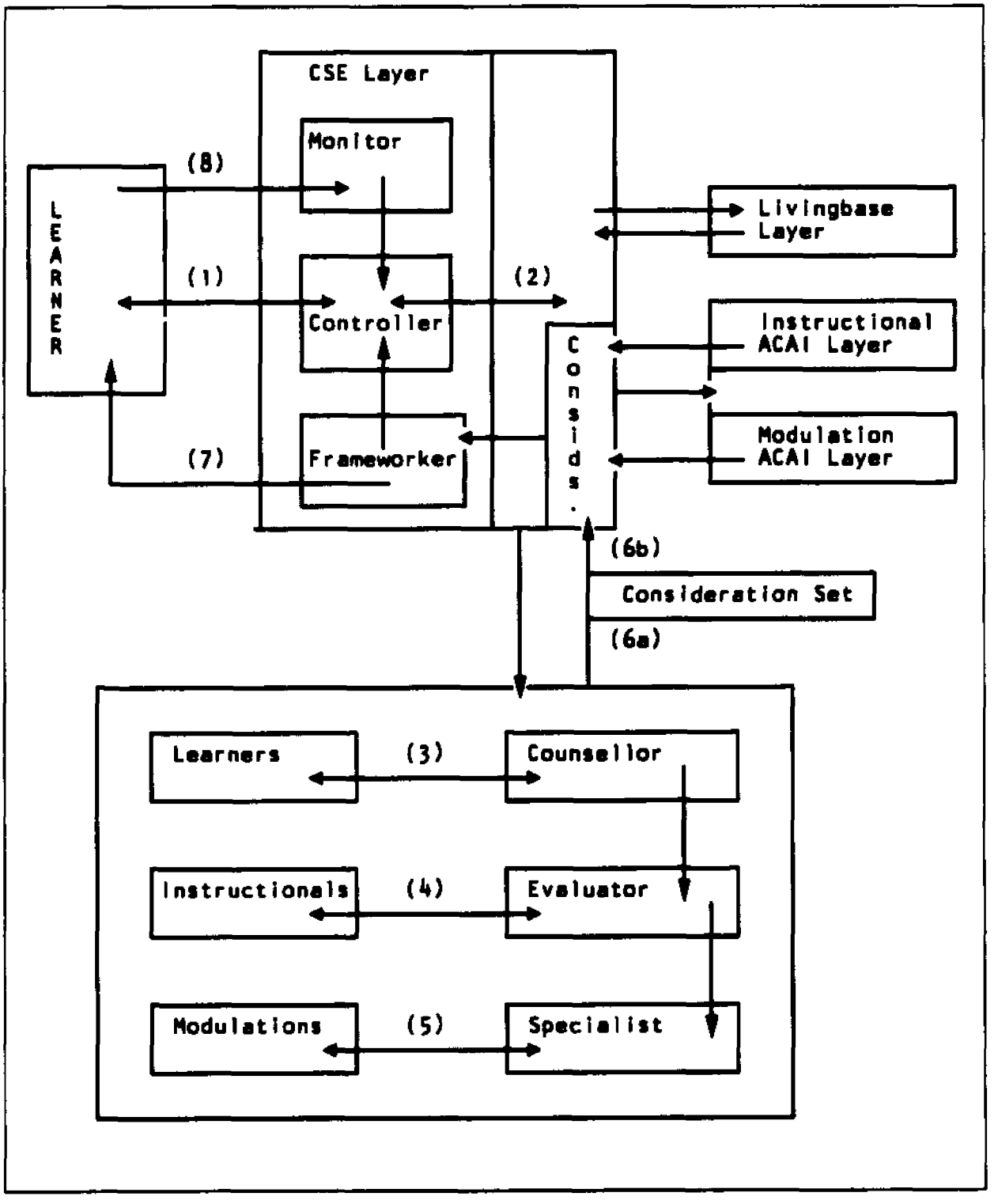

Figure 4. E³ Process Flow

The Educational Engine

It is convenient to view each active session of E^3 as a series of cycles of the driving mechanism that runs it. The active mechanism which comprises the activities assigned to each platform and drives all their processes is referred to as the *Educational Engine*. The Educational Engine works in cycles corresponding closely (in this interpretation) to the cycles of the inference mechanism that implements it, but with some added detail.

The illustration that follows shows the flow of control during the execution of one cycle. The steps are as follows.

A session is requested by a learner entering the environment.

1. The Controller requests the initiation of a new cycle.

   The set of considerations, instructionals and modulations (C,I,M) set is gradually built in steps 2, 3 and 4.

2. The C subset is effectively created by the Counsellor.

3. The I subset is effectively created by the Evaluator.

4. The M subset is effectively created by the Specialist.

5. The Consideration Set (set of all instantiations) is completed.

6. The Consideration Set is used to formulate the application of a *real* ACAI Component.

7. The Frameworker works with the selected component.

8. The Monitor receives information from the active ACAI at built-in break-points.

   Based on the information received the Monitor has the following options:

   - transfer control back to the Consideration Set effectively switching sets;
   - continue with the current ACAI until it is exhausted; or
   - return control back to the Controller effectively starting a new cycle.

9. The Controller (at this point) has the following options:

   - store pertinent information, terminate the cycle and start a new cycle; or
   - store pertinent information and terminate the cycle and the session.

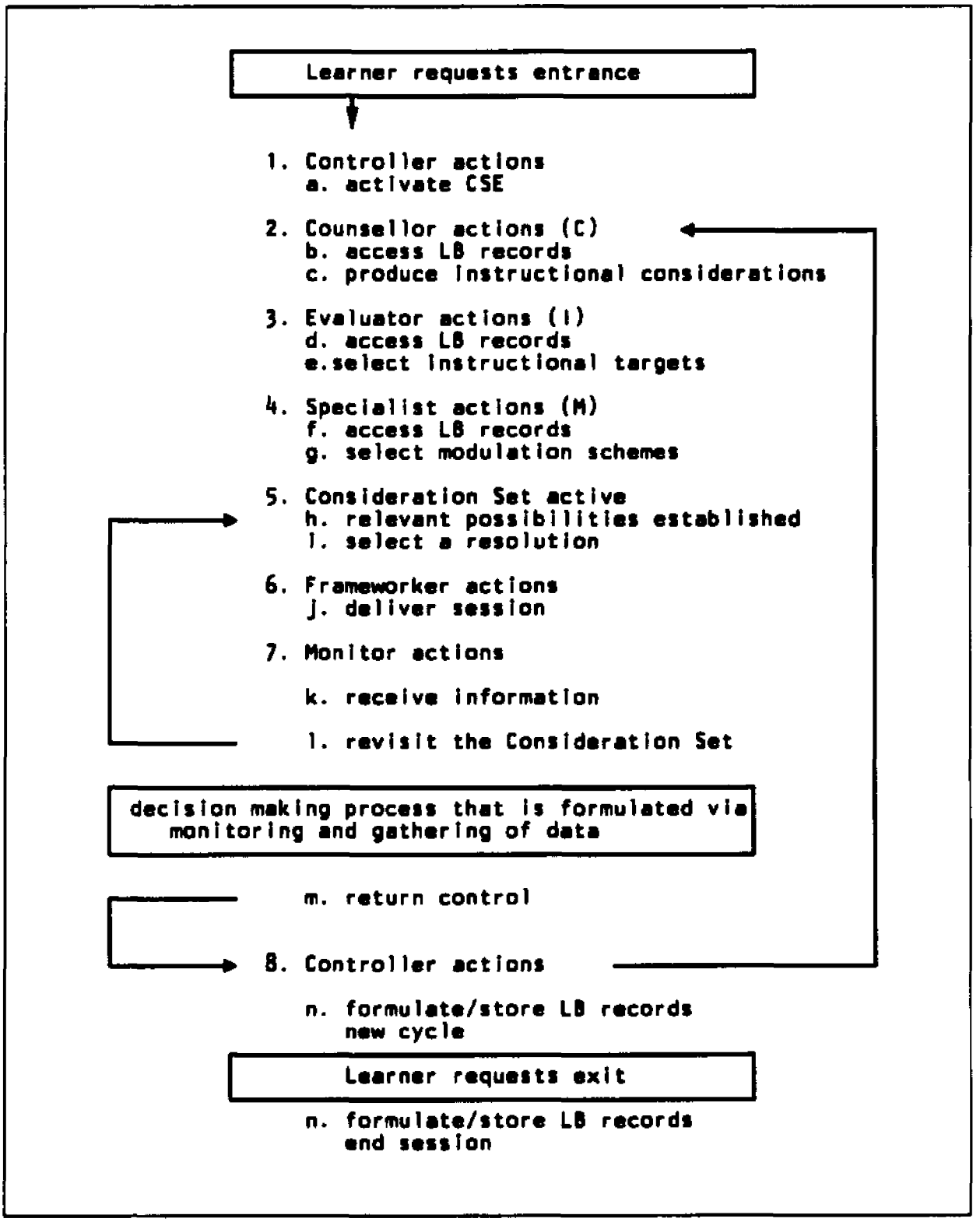

Figure 5. The E³ Cycle

The Consideration Set

Earlier, inferencing was mentioned. Inferencing in a knowledge-based system is carried out by a separate piece of software called the *inference engine.* See Section 2, "Necessary Information" on page 5 for more information on this topic. The inference engine recognizes conditions that have been satisfied and triggers the execution of actions associated with those conditions. This is typically done by means of some efficient pattern matching algorithm. One prominent example is the *RETE Network*, (Forgy, 82).

A simple illustration follows.

```
When Workpage 1 in Workbook A
has been successfully completed  =>  Advance to Workpage 2
```

In some cases more than one condition is satisfied, then more than one action is potentially available for execution. The inference engine typically makes a decision on which action to select for execution from among all the condition/action pairs that have been satisfied, more specifically from the *conflict set.* This is referred to as *conflict resolution.*

Within the mechanisms of the inference engine a set of heuristics is typically employed to make that decision. One criteria for selection might be recency of

condition satisfaction. That means that the last condition satisfied will have its action portion executed.

This approach is quite satisfactory for many applications. For this particular application a Consideration Set, effectively equal in content to a Conflict Set, becomes more useful. The major difference is in that competing condition/action pairs are not discarded, but rather saved as sets of possibilities, in this case *considerations*, *instructionals* and corresponding *modulations* for those instructionals.

The scheme is sometimes informally referred to as *RETE save*, which simply means storing some place the current contents of the conflict set. Ideally the tool (knowledge representation language) used to implement the application should contain constructs to facilitate RETE save. However if this facility is not available additional programming techniques imbedded in the application itself can be employed to produce the same effect.

The operational logistics deeper into the CSE would implicate the use of such added techniques to create the Consideration Set. A gating mechanism that makes use of *flags* within the patterns can be effective in having control over what conditions should not be discarded. The use of *negative patterns* can also be used to activate and deactivate conditions. The idea is to operate on the current set of conditions and actions (conflict set) for as long as the set is potentially useful.

The concept, is that all currently satisfied conditions are relevant to the learner. These should be available for consideration before changes are caused by interaction with the system that may affect the contents of working memory, consequently affecting the active educational path.

The creation of a Consideration Set is an attempt to recognize the importance of continuity in Education, as is the importance of flexibility recognized. So it is in a way a leverage factor within a system that is highly responsive to changes. Therefore a jump to a totally different learning path will not be made too quickly, or at least not until some attempts have been made to exhaust the current path.

Associated ACAI Scheme

As seen earlier, one by-product of the CSE process is a series of *considerations.* These are selected from encoded action/goal pairs. These considerations are present in the Consideration Set and serve to initiate the process of combining modules propagating to the *instructionals*, then to the *modulations.*

The satisfied conditions in the Consideration Set represents the sum total of all current CSE processes, then the actions represents the ACAI processes that will follow. The CSE processes establish the conditions that need to be satisfied for this session. The ACAI processes satisfy them.

```
CSE
-Condition: what are the educational goals => consideration
           what tools to use => instructional
           how to use them => modulations

ACAI
Action: how to combine a given consideration with a given
        instructional and its corresponding modulations
```

For every set of considerations selected, there is a set of instructionals that fit those considerations. And, for every set of instructionals selected for those considerations there is a set of available modulations capable of refining or enhancing their application. Considerations have instructionals associated with them and instructionals have modulations associated with them. The total given (selected) set must be linked in some fashion that reflects the way in which they were originally designed to work together. A *linked list* scheme is enough to effect the desired relationships. But the CSE must chose carefully. All (C,I,M) sets selected to become part of the Consideration Set should be compatible, all possible combinations must be associated with the current educational path.

In the illustration that follows considerations C(1 to n) are associated (can accept) instructionals I(1 to n) in such a way that the combinations are compatible with the current goals. The same is true for M and I. The selections are propagated from C to I to M. The entire set is useful during the session, but it may not be interchangeable. That is, a single C may not necessarily combine with every single I, and every I may not accept every M in the set.

The possibilities for modulation depend on the entire available set and its interrelationships. It is therefore possible to modulate sessions by altering the I to M choice. And, it is possible to do this several times until the correct (available) modulation has been found. Likewise, but for different reasons, it is possible to change the current session by selecting a different I with its corresponding propagating effect on the M set. The most dramatic change within the set would be accomplished by selecting a new C. In this manner each cycle of the Educational Engine is more efficiently utilized. But more importantly educational possibilities that fulfill the current educational path are exhausted in an orderly and organized manner.

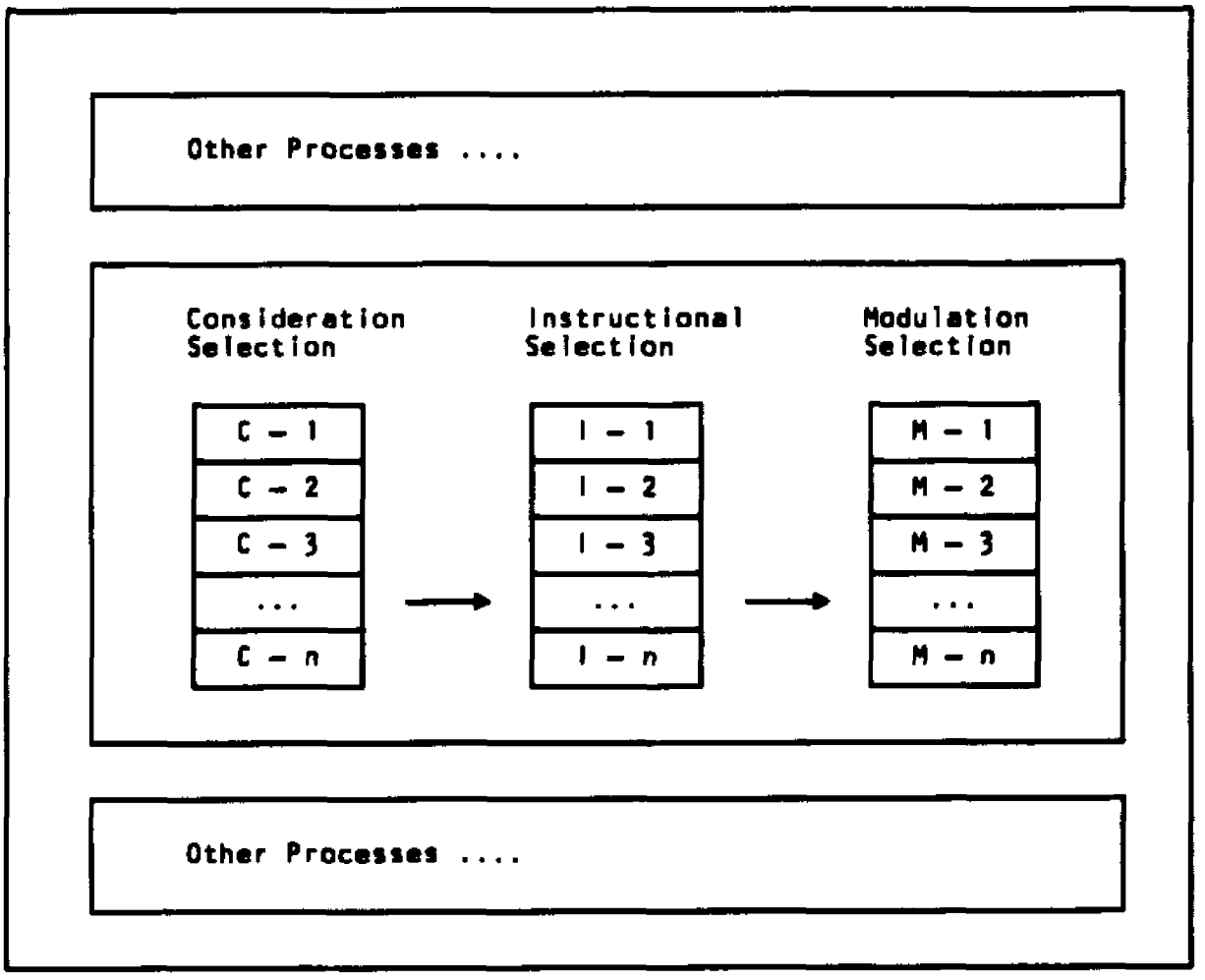

Figure 6. The Consideration Set in ACAI Processes

Section 15. ACAI Process Scheme

The ACAI approach combined with the CSE knowledge-based approach form a more robust philosophy of action for computer-based education. The reasons for making the CAI component of E^3 more adaptive, by breaking each sub-component into its instructional and modulations aspects are twofold.

First as has been explained previously, it provides high adaptability to single students supporting the concept of individual education. It does however also helps supports the concept of *sustained individual education.*

One of the shortcomings of the educational CSE process, as it takes place in the schools, is that it drags on, as IEP(s) must be provided in an ongoing basis. The three year review period in itself says that. On the other hand the review process has as one of its goal to assess the possibility of main-streaming, which supports a contrasting approach, *group learning*. E^3 is designed to sustain IEP production for conceptually endless time frames and for unlimited groups of students. So that group learning, in the sense of employing an approach to education that simultaneously services many students; and individual learning, that attempts to service each student according to current needs, can be both realized concurrently.

One way to do this is to have a large variety of ACAI software (for many subjects, levels, teaching strategies). But *large* in this case maybe the same as

intractable. To compensate in some way and to enhance the available variety, ACAIs must be designed to be utilitarian.

It has been suggested that one or more consideration may result from CSE processing. And, that a corresponding set of one or more instructional and its associated modulations may be available to satisfy those considerations. Once a consideration, instructional and modulations set has been isolated those possibilities are tied to individual education and the preparation of a single *flexible* IEP. However, it is possible to design ACAI software in such a way that there is a super-set of considerations, instructionals and modulations that can be combined interchangeably for specific purposes. These combinations would be identified by the CSE from the information about then contained in the LB.

This added feature would increase the complexity of implementation for those ACAI modules that utilize it by an order of magnitude, but with it the power of the ACAI approach would also increase.

Adaptability Scheme

In conclusion, the concept of retaining an active Consideration Set makes:

- inference cycles more proficient;

but also facilitates ACAI processes that make:

- educational paths more stable,
- IEP(s) more flexible, and
- instructional software potentially more useful.

Beyond that, it yields another form of adaptability that relates to a a different problem. The problem considered now is how to make instructional software capable of sustaining educational services across student population, with the same scheme, for sustained periods of times. The goal is *continuity of individualized instruction.* The scenario that follows does not represent an example, but serves to illustrates the idea in conceptual form. The parameters of the illustration (skills being taught, mastery levels, teaching approaches) are not necessarily real and have been constructed only to drive a point concerning ACAI software design.

Continuous Educational Services

Case 1: Reading/Writing software for levels 1 through 2

Consider instructional software designed to teach reading level one students, by introducing them to the phonemes present in the English Language. Consider that while this software has been effective in many cases, Educators may shy away from using it in cases that require special teaching approaches. In such cases and according to some given methodology, which may be appealing to some educators, a presentation that reorders the phonemes may be more effective.

With initial extra effort on the part of the software implementer, the instructional speculated above may be made useful and acceptable to more educators, including those who prefers to adhere to the reordering of phonemes approach. Here the same instructional concept is modulated to meet two *instructional approaches*. Recall that earlier modulations were used as examples of possible forms of *instructional strategies*, but implicitly under the same instructional approach. In this case the same instructional software would then be used for more than one instructional approach (preference) depending on the situation.

Design and implementation of this kind would require additional study and extra effort, but could increase use and acceptance by many orders of magnitude, depending on how many strategies and teaching approaches are suitable for incorporation. Conceivably, this kind of software development can be done on a gradual basis, as a form migration or enhancement path for existing instructional software.

Case 2: Reading/Writing software for levels 3 through 5

Consider next that instructional software exists that has been designed particularly for learners who profit from special phoneme reordering approaches. This software adheres to those methodologies exemplified by the reordering, but lacks the flexibility for reasons equal but opposite Case 1. The same tactic as in Case 1, (re-ordering of phonemes) perhaps combined with other methodologies (speed of presentation and varying repetition factors on drills) would again increase the usefulness of the given instructional as well as teacher acceptance.

Case 3: Reading/Writing software for levels 6 through 12

Consider now that older students can benefit from given sets of instructionals designed to foster reading comprehension. These can be presented in agreed upon formats which may be considered conventional, or they may presented employing repetition, word isolation and other special modulations which are more suitable for students with specific needs.

Again the same material may be equally useful for a larger set of students via modulation. Multiplying the methodology used to convey the material also multiplies the material to some extend, amplifying its power.

Implications of Software Adaptability

Three hypothetical cases of instructional software were presented as they would be operational under E^3, showing speculatively how their usability could be extended across populations of students. The approach, however, has even deeper implications.

If the software hypothesized above existed, its use would provide more of the needs of the students than the same software independently used outside of E^3. Hopefully the same software would gain more wide acceptance by students and educators. Without favoring or patronizing anyone teaching technique and in a

non-competitive way, all can conceptually coexist in E^3. The educator via the E^3 System would have the ability to employ favorite and/or more effective techniques in any given case. This would simply be a function of allowing certain instructional/modulation sets to run under the environment.

This can be viewed as an automatic form of curriculum planning, with the educator as the re-enforcer in the activity. Another possibility would be not to limit the kinds of instructional/ modulation sets, but rather to prefer certain ones for, or not for, specific students. These are implementation details that can be manipulated by externally manipulating the databases.

Additionally, as mentioned earlier, E^3 would have the power to sustain educational services not only across student population but for prolonged time frames. The students, conceptually depicted in the above cases could receive educational services that meet their needs from level 1 to level 12, with instructionals that come from various sources, but that work together. Educational paths would not stop or deteriorate and become more and more general as students advance with age and sophistication. This is one important difference between this model and previous ones, continuity of individual education is part of the original purpose.

As a final comment, the hypothetical examples above are meant to be simplistic representations and do not capture, nor try to capture the complexity of the real strategies needed in those cases. The point of the illustration is that the same

ACAI adaptability scheme used to tune in to the individual needs of the learner is used to amplify the usefulness of the instructional software prolonging service-ability.

Section 16. Behind the E^3 Architecture

The specific reasons why this particular architecture was chosen become apparent when examining the real-life processes it tries to represent. Evidence drawn-out from the literature helps to validate some aspects of it. But, there remain other issues to be examined and supported that have to do with Computer Science practice and theory. Those are now examined, for the careful consideration they were given during the design of the E^3 Model.

Isolation of Functionality

As an elaborate system design task, E^3 requires the maximum amount of isolation of functionality that is possible for the application. Partitioning this system into its most functional components simplifies not only the programming task but also understanding of the problem. This isolation approach seems intuitively logical, since the general domain requires the application of various kinds of highly specialized knowledge which are not generally common to one person. The applied knowledge of a group of dedicated and experienced people is combined to produce the desired best fit program for a student in the educational setting. Each contributor bring distinct aspects into the process.

By combining the elements of knowledge and experience and then conceptually modifying the results to fit current needs, the special people involved in this

process are able to arrive at original solutions. There are many variations of the established procedures, but in general they include:

- setting educational objectives,
- testing and diagnosis,
- selecting materials,
- arriving at remedial prescriptions, and
- post-testing and diagnosis.

And many iterations of this are usually necessary, as is evident by the cyclic nature of the special education process.

In order to best capture all the knowledge needed to carry out the tasks attempted by the system, both the knowledge representation approach taken and the tools later utilized to implement it are now considered. This system is designed within a global picture of:

- what is needed and wanted (educational considerations), and
- how to use technology to achieve those goals (computer science considerations).

A Previous Example

Knowledge-based systems do not easily capture the essence of what is needed for a system such as this one. In particular with highly complex systems, as it was demonstrated with the results of YES/MVS I (Griesmer *et al.*, 84) and some

deficiencies in the approach must be overcome. In that particular experience isolating functionality to ease representation was explained as functions of the following tasks.

Isolation of Expert System Interfaces

To promote the separation of independent tasks. Those that are more closely related to management of educational services than to actually administering them, would be an example in this case.

Isolation of Subdomains

To better organize knowledge and its logical interactions. In this case one would tend to separate the roles played by the teacher, from those of the psychologist, principal, special resources teacher, etc. into subdomains. These are logically distinct in the educational environment and should remain so in for this model.

Isolation of Particular Operational Knowledge

In this case, and learned the lessons with follow-up studies of YES/MVS I (Cruise *et al.*, 86), the architecture of E^3 becomes itself an example of isolation of functionality.

A Procedural Knowledge-Base

The above title, paradoxical as it may seem, conveys relevant ideas put forth by English scholars. In addition it is not only desirable, but entirely possible to construct such a thing, even when using only a programming language. As a way of explaining, let us consider first how a *procedural knowledge-base* can be constructed in English.

Consider two processor communicating with each other. These processors are human beings trying to convey information that:

- must be transferred,
- must be understood,
- must be processed (in some form).

Consider next that these activities must be carried out in written form, that is, the processors (human beings) must rely on abstract symbols to perform their respective tasks.

Those are fundamental concepts of human communication via written language. The study of such activity outdates that of designing computer programming constructs by many centuries. The lessons that have come down from the Masters suggest simplicity and consistency of style. One particular example that comes close to this discussion can be found in Joseph William's *Style - 10*

Lessons in Clarity and Grace, a model worth applying to the art of designing software.

The Teachings of the Masters

The use of language as related to the way people think, is the focal issue in that book. William suggests that a written paragraph should start with a sentence that introduces the ideas being communicated in it, and end with a sentence that summarizes (logically concludes) the paragraph. The paragraph that follows, then should start with a sentence that carries through the message within the concluding sentence of the last paragraph and embodies the ideas currently being presented. The process is repeated with every paragraph thus relating all the ideas logically and sequentially, one step at a time.

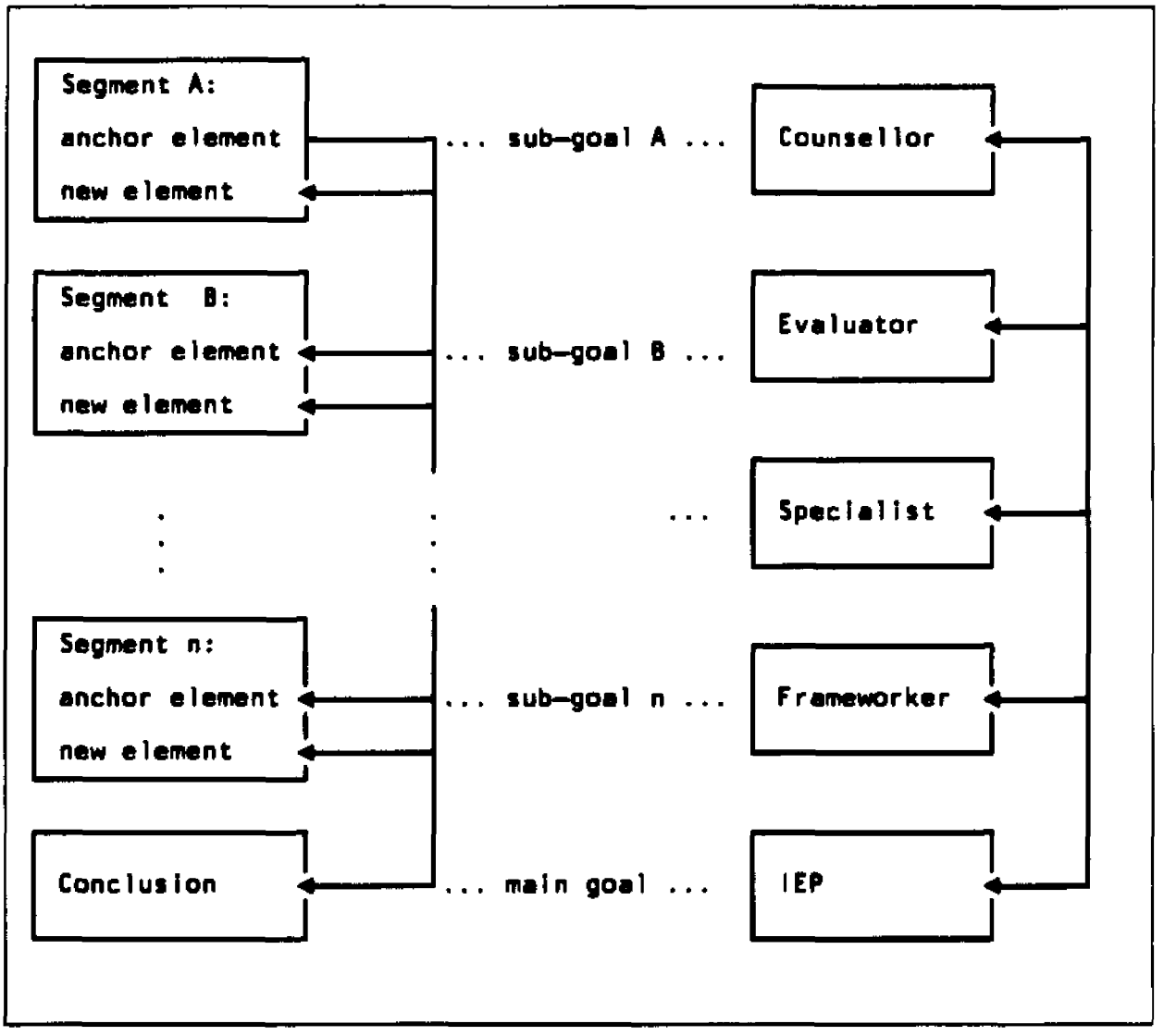

Figure 7. Natural Sequence

Procedural Style Communication in Systems Design

The process represented above can be viewed as a well structure system for explaining a process or for capturing the process. It can also be viewed as a goal satisfying process. Satisfying the main goal of the process (concluding or

completing the process), is a function of satisfying (explaining or capturing) all of the Subgoals (A, B ...n). This goal satisfying process should to be guided by laws that:

- control its behavior,
- can be expressed with language (represented),
- have a hierarchical structure.

The implications become even more profound when considering:

- clarity,
- coherence,
- emphasis, and
- conciseness.

If these are considered functions of decision making.

The preservation of conditions is yet another function of the process whereby future states are concerned. In the E^3 Model the decisions made up front heavily weight on the conclusion, because they are preserved and modified over time.

The E^3 Model attempts to embody all of these principles in its design.

A Message to Consider

The not unfamiliar lesson is that procedural communication and its related concept of procedural programming, that works for human beings and systems as well, perhaps has not been emulated with the declarative style sometimes advocated in recent practice. This does not imply that the declarative style is not useful, but rather that it may not be better than the earlier approaches. However in order to take advantage of both, not only is a combination of both programming constructs within a single system suggested, but a combination of the philosophies behind the approaches as well. This is what is meant the term procedural knowledge-base.

Emulating the Human Processor

In the hypothetical human-processor interaction example, some important considerations were left out. Information specifications in one processor (human mind) when conveyed to a second processor may not capture the meaning (reasoning process). Therefore the procedural flow does not guarantee covert logical analysis, and, the desired results may not be obtained (understanding). Some intentional form of procedural expression is required. This requirement has not just come down from English scholars, but from computer scientists as well who have long advocated for structure and modularity. In programming terms that could, among other things, mean that negative-default fall-thrus are undesirable. (*The Psychology of Computer Programming* Weinberg, 71) and (*A Structured*

Approach to Programming, Hughes and Michtom, 77). Implemented in its best form, structure programming reduces ambiguities, and modularity reduces the distance between related structures.

It is possible, with software, to be clear, concise, effective. It is possible via software to communicate ideas and knowledge, if no more perfectly than via oral communication. These considerations of style are important when designing an educational system, because teaching and learning are intuitively approached by educators and students, as processes intimately related to the acquisition of knowledge by means of recombining the elements of language. But even more formally, learning has been viewed as a knowledge communication process coupled with a knowledge compilation process (Anderson, 85).

These views must be taken seriously and the form and style of the system used to support the learning and teaching processes, must then be a well balanced combination of both structure and representation. Not only need the structure be carefully planned, as it has been for the E^3 Model, but the design must constrain any possibility of representing valued knowledge in any order, hoping that by means of some low level matching mechanism their functionally and applicability will be somehow extracted when necessary, as is often done with expert systems.

In addition to the above considerations an even more detailed formalism for knowledge representation is attempted, requiring the underlying existence of a

learner model, or target of the processes being represented. The knowledge being represented is then itself a model of the process of applying that knowledge. These are considerations that affect the design of the modeling scheme.

Section 17. Modeling Scheme

In addition to the previous considerations an even more detailed formalism for knowledge representation is attempted, requiring the underlying existence of a learner model, or target of the processes being represented. The knowledge being represented is then itself a model of the process of applying that knowledge. These are considerations that affected the design of the modeling scheme.

To capture the effect that knowledge application has on the target problem or target for the problem, a deeper form of representation is necessary. This is not completely a function of the architecture and needs to be carried out via some other tactic. A modeling scheme that represent not only the object being modeled, but also the changes that operate upon that object is employed.

Implicit in the knowledge-base of the CSE there is a scheme to represent the student. The representation, or model, will be as simple or as sophisticated as the knowledge contained in the CSE components and supplemented with the information in the LB components. This model then grows in sophistication as the hosting E^3 does (with use and enhancements).

The process and the model are completely transparent and newly created with each session and for each student. Each time a user interfaces, the model changes. This is so because:

- the data contained in the LB changes each time a user interfaces, thus the model changes - (the information mapped to the knowledge-base is different), and

- the CSE knowledge applied may be selected differently with each interface, thus the model changes - (the mappings are different).

The basic concept beneath the modeling scheme is to implement a mechanism that supports a series of enhancing learning transitions, from the most fundamental to the most complex that the current E^3 is capable of providing. More precisely, the system is committed to promoting the student along educational paths, supporting learning transitions, with relevant fine-grained learning situations, that are procedural and additive. The model is designed for the anticipation of learning impasses and for adjustment of the learning situation accordingly.

The educational paths should follow increasingly sophisticated educational levels. A transition mechanism is then required that regulates those transitions while at the same time characterizes human developmental and learning processes. In this case the learning processes are seen as specific forms of skills development, or human knowledge acquisition.

Progression in Knowledge Representation

In order to be able to represent this form of modeling with software a very soft progression of knowledge representation is necessary, refining control for capturing the model.

A progression can be viewed as a set of *pictures on the left* coming closer to a set of *images on the right*, attempting to trace movement along the learning paths. One way to represent this idea (in AI terms) is with patterns that progress along goals. But adding, that the total model must come closer to this pattern/goal set (as a transition set) in achieving a given main goal. The concept can be illustrated as follows:

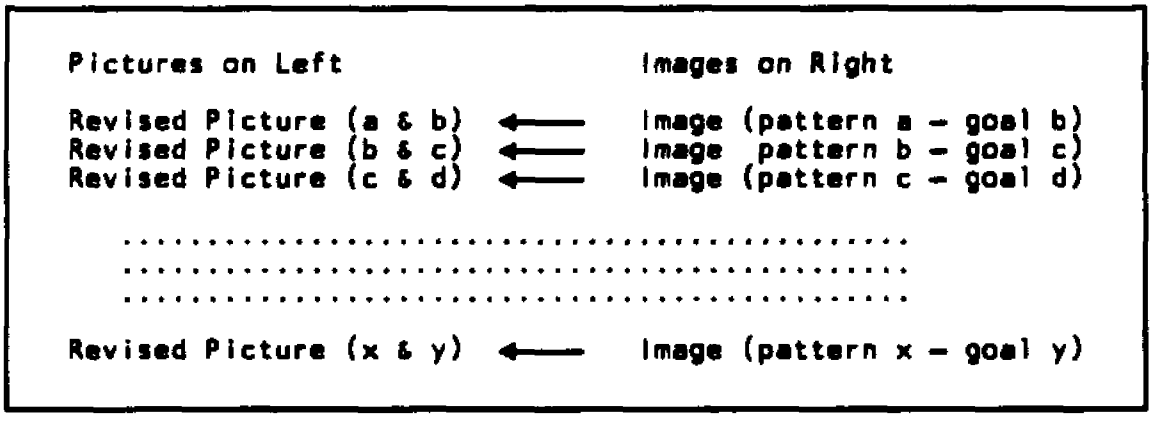

Figure 8. Goal Achieving Process

For a certain set of patterns (pictures) there is a corresponding set of goals capable of creating a new picture. For instance pattern/goal pair a/b is capable

of creating a new or more specifically a newly revised, picture that contains both a & b. That is, there is a set of given conditions (expressed in patterns) and actions (expressed as goals), that when taken normally fulfill those conditions.

If for instance:

- the condition is *a*, and
- the action is *b*.

Successful completion of this transition from a to b would result in a new picture that:

- contains *a*,
- but now also contains *b*.

therefore *b* is now an active element in the revised picture and capable of modifying the set of new operational conditions.

In knowledge-based systems changes in working memory have the potential to create other changes. It is in fact the case that (for knowledge-based systems in general) unless changes are propagated to working memory nothing can possibly happen and the system could enter an infinite loop.

The technology is appropriately utilized here, with the underlying mission to move along learning transitions, very gradually, trailing them.

The pattern/goal pairs on the right then represent, in concept and for this application, all the things students may possibly ever learn. Thus it is an infinite set. The assumption is that sophistication in modeling is dynamic, the more pattern/goal sets that are added to the environment the more powerful the modeling scheme becomes, and implicitly so does the system.

However since only a handful of patterns are applicable at any given time (candidate rules in the rule firing process), the potentially infinite model is reduced to a relevant model. That is, only that knowledge and information that the learner needs in order to be able to tackle the concrete instructional/problem solving situation at hand.

The scheme is to break down overall, the object modeled (in this case the student) into its most relevant characteristics in the same way that instructional facilities were broken down into its most functional pieces. Then use the model by simply getting one picture closer to another potential picture and goal set (new image) that apply to the current situation.

The process starts with a picture (on the left) which needs to be revised according to some pattern/goal set with an approachable image (on the right). This new image is blended on the left as a form of revision.

The difference between the two is the set of changes to the model, or the transition delta. Merging of changes is required in order to move to a revised picture,

effectively a new picture, by a process of propagating changes. The current version of the model then would correspond to the changes between the current condition and the current action, in order to progress to a new version of itself. The model responds to stimuli from conditions and actions and thus modifies itself.

This process takes place in a very systematic manner. An attempt is made to reduce the difference between all relevant instruction that is available to the system and that which is already part of the student model. That is, the difference between what the student already knows and what the student could now learn. The student model is modified (potentially) each time by one task.

Having intuitively presented the scheme, it can now be defined using some appropriate nomenclature for subsequent reference.

Scheme for a Living Model

The *Living Model* (LM) is composed of a *Revision Model* (RM) which represents very explicitly the current *Picture* of the student. This Picture is revised as the student uses the system and moves along learning paths. Thus it is called revision. It is revised by extracting *Images* from a more encompassing model called the *Approachable Model* (AM), signifying that the student can get there. The *Delta Model* (DM) is the route for transition, the learning path, thus it represents the things the student learns at given times.

Particulars of the Modeling scheme are as follows.

- The process is repeated moving closer (approaching) the the AM by virtue of progressing along a set of images taken from it.

- The deltas are neither stored nor read, they are evaluated just prior to the merge, as a side effect of the process. These, however, can be subsequently recorded as an educational audit trail mechanism.

- Current deltas are equivalent to cause and effect interplay between the current version or picture of the RM and the AM images as that the student creates by interacting with the environment.

- Anything that changes an instance variable (e.g. changing the content values of attributes) is considered a delta generating side effect.

- The desired direction of movement is that which more closely encompasses the AM, however, evaluation of the current delta may move the RM to an old version of itself, which contained less of the AM.

- This is considered theoretically possible for this model, and corresponds to the overall concept of *forgetting*. or not being able to locate *the* most recent picture(s). Notice that the function provided for delta, implies sub-functions for augmenting and demising, implicitly in the definition of the DM.

- A sub-function for demise is defined in terms of a change that appears to have removed images from the RM. That is, a set of images that was recorded as having been present (part of the existing picture), now seem to

be unclear, forgotten. The picture is different, but the images are not removed, rather obscured (flagged). There is a loose implied correspondence for the need to *review* these unclear images in the presence of a demising merge. It is important to understand that this makes the ability to asses the need for review inherent to the method for representation.

This modeling scheme represents the student in such a way that:

- the profile of the representation is always fresh;
- the expectations for the representation are neither set for, nor directed to any given specific picture;
- the concept of learning has been incorporated;
- the concept of forgetting has been incorporated;
- the model is completely pro-active in terms of the propagation of deltas.

The idea of a revision model in connection with the use of systematically designed educational materials was studied in some detail in Rayner (72). However, that particular model targeted revisions to the instructional materials. Those revisions are also "project specific" (instructional materials specific) and for each project such revision criteria would have to be derived. One interesting conclusion from that report is that "CMI would appear to be a reasonable vehicle for investigating a revision model". The perspective of E^3 is different and corresponds to transitions of the learner, not of the learning materials.

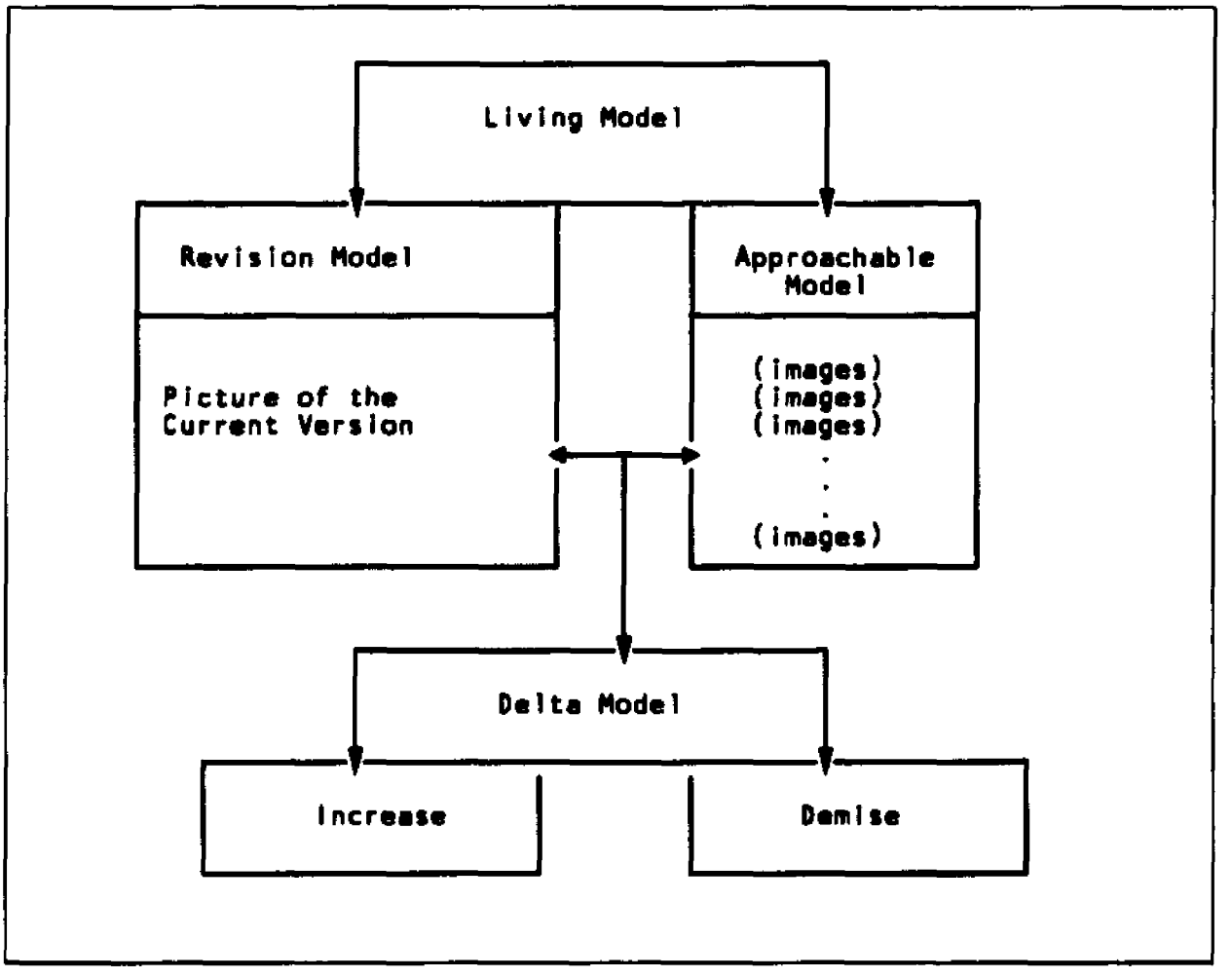

Figure 9. Living Model

Modeling Scheme Definitions

The modeling scheme employed here is completely dynamic and conceivably total, representing not just the students, but various pictures of them, as they are, as they were and as they could be; and hopefully reality will be escalated in that sequence, but that is not a requirement. Also, there are no existing repres-

entations of students in terms of specific goals, but the goals are matinal and formulated in part by the model itself.

Definitions associated with the modeling scheme are the following.

Living Model is a subjective computer representation (of an E^3 learner) capable of modifying itself to better portray the object being modeled. It includes three characteristic pictures of the model, each in itself a different and independent aspect of the model or sub-models:

- the Revision Model,
- the Delta Model, and
- the Approachable Model.

The Living Model portrays movement along respective characteristics in all three.

Revision Model is a current picture of the student.

Delta Model is an applicable difference between the Revision Model and the Approachable model.

Approachable Model is an infinite set of images. These images are used to formulate specific deltas. The Approachable Model is a conceptualization of all things that a student may learn.

Image is a representation of something that can be learned by a student. It could be interpreted as goals. Images are used as targets by the Revision Model.

Picture exists in the Revision Model. Also the given set of pictures existing in the Revision Model.

The definition of a picture of the Approachable Model is also conceptually possible, but would implicate that there exist images for the Approachable Model to use as targets. The mechanism for modifying the then potential pictures in the Approachable Model would have be formulated. A possible avenue for creating such a mechanism would implicate the intervention of the Oracle which is an optional component of the CSE. In that case and at that level of performance the Approachable Model would then be referred to as made up of pictures of images and be capable of modeling after new images.

Model Version is a current composite set of pictures of the Revision Model or Approachable Model.

Increase Factor is a transition in the Delta Model to merge images from the Approachable Model into the Revision Model.

Demise Factor is a transition in the Delta Model causing images to be obscured in the Revision Model.

Learning Activity is the process of increasing the Revision Model with images from the Approachable Model.

Forgetting Activity is the process of obscuring images in the Revision Model potentially necessitating review.

Reviewing Activity is the process of clearing images in the Revision Model. (A mechanism for review need not be formulated. It requires some reprocessing of images as if they were first being added to the Revision Model. These previous images are known to the LB).

Recording Changes

Trailing the progression of the model is a function of reporting deltas. Recording each new version of the model is possible by saving pictures. The model creates a recorder since the deltas provide histories of the student. Therefore implicit in the scheme there is a machinery to extrapolate reports, and it is not necessary to device other schemes for that purpose.

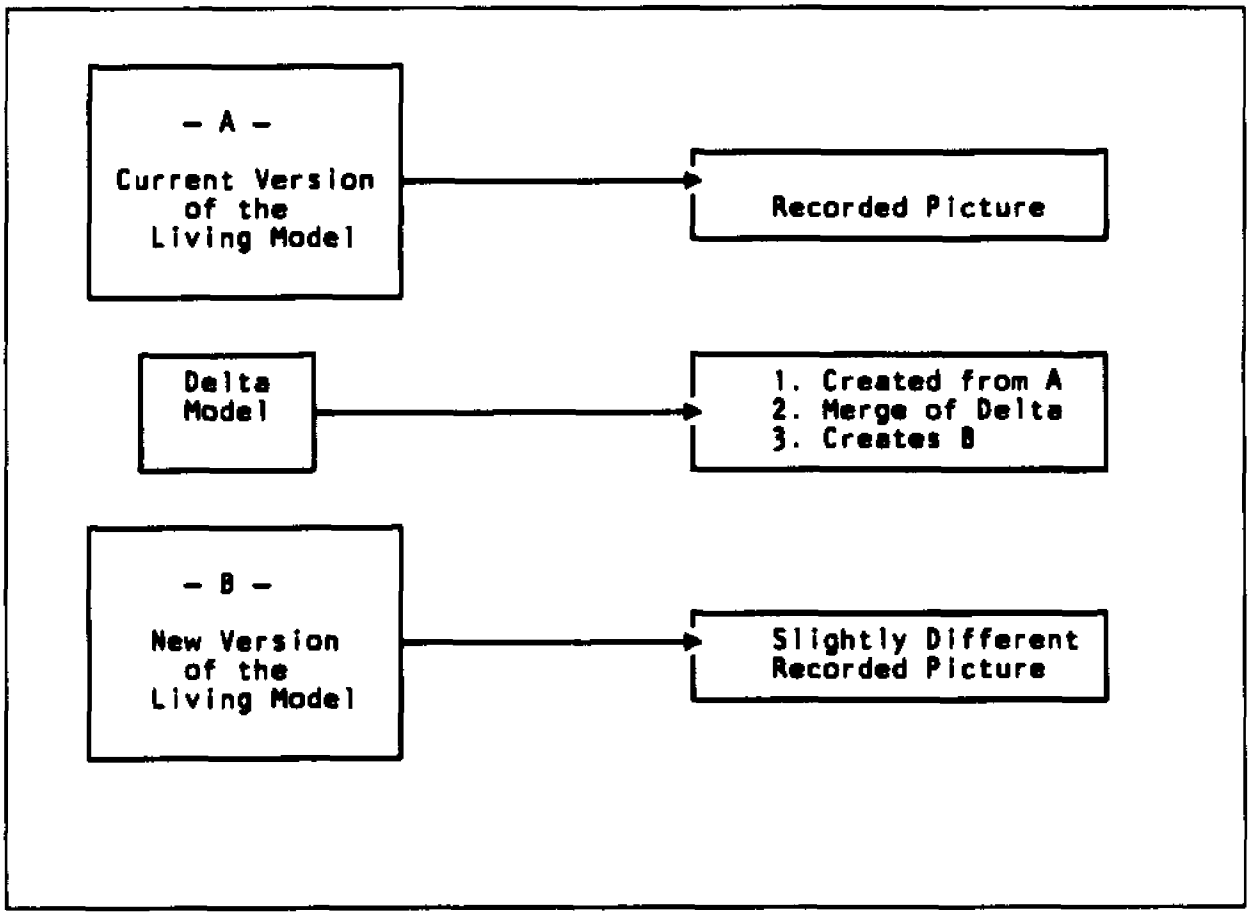

Figure 10. Recording Changes

A fine point that must be explicitly emphasized is that the changes that are taking place are useful as information, but are not the criteria used for formulating other changes. So in essence s student's educational program is not based on his current track record but on his current need and abilities. Review, however, would take place by default since new educational goals when assessed, typically build on previously met educational goals. If these have not been met, or are not now present the system is forced to formulated new goals accordingly, and include a new attempt to meet previous goals if that is the case.

Part Five: E^3 Mathematics

Section 18. Topic Description

A scenario, representative of a working subset of the E^3 Model is presented. This is done to convey more precisely the way the system works. It is also an abbreviated road map for transition from conventional instruction of a topic, to its E^3 counterpart. This scenario can be used as an example to develop other scenarios and preliminary specifications.

These are points to keep in mind however.

- The focus is on the system itself, not on the example being presented.
- The criteria for demonstration is then that which can clarify conceptual coherence for a small subset of instructional paths.
- The goal is to illustrate how the system would work, not how effective (educationally) the actual example is.
- The purpose is to demonstrate that the system is doable, and using technology that is available.

The Topic and its Importance

The area chosen for this scenario is Number Theory. Number Theory explores the relationships among counting numbers and their properties. Concepts

include odd and even numbers, greatest common factor, and least common multiple. These are difficult to teach because they require understanding of the taught process necessary to work with patterns, conjectures and validation of conjectures. The topic is nontrivial. According to the NCTM curriculum standards (*Commission on Standards for Schools Mathematics*, 1987), looking for patterns is the essence of inductive reasoning. Within this area factoring has been selected for demonstration because of its important role in the development of mathematics foundations, including divisibility, common denominators, rational numbers (common fractions) and the Fundamental Theorem of Arithmetic or Unique Factorization Theorem (a distinguishing feature of our number system, that every composite number can be expressed as a product of its prime.

The topic is presented as follows:

- factors,
- common factors, and
- greatest common factors.

An educational path is traced that requires as a prerequisite:

- understanding of multiplication concepts;

and culminates in understanding of:

- characteristics of numbers,
- relationships of numbers,
- rules of divisibility,

- some abstract mathematical principles (e.g. commutative law).

The topic is an essential learning objective. It covers a range of developmental levels of the following scope and sequence outlined below.

- Fourth Grade
  - common multiples
  - understand multiples
  - odd and even numbers
- Fifth Grade
  - find patterns in number sequences
  - greatest common factor
  - find least common multiple
  - prime and composite numbers
  - use factor trees
- Sixth Grade
  - prime and composite numbers
  - prime factorization
  - divisibility rules
  - least common multiple
  - least common denominator

It is also adequate for demonstrating a range of instructional strategies targeted at specific or combinations of teaching strategies and learning preferences as outlined below.

- Global strategies

  - can the student see it - visual recognition
  - can the student say it - auditory recognition
  - can the student write it - abstract understanding

The sequence thus bridges the concept from the concrete to the symbolic and abstract. Only some of the key aspects in this topic are used in this scenario. For those simple examples are given on how all the pieces would be put together.

Section 19. Platform and Component Levels

Platforms

The topic chosen has been presented above in its educational importance. More significantly it has already been broken down into discrete elements that fit into the E^3 Model within sound educational practice. Specific aspect of the topic are now mapped to corresponding E^3 components and sub-components.

At the platform level this is done as outlined below.

1. CSE Platform

    - knowledge necessary to use the information needed to match the student with the instructional topic and teaching strategy

2. ACAI Platform

    - Number Theory (Factoring) - instructional software
    - Number Theory (Factoring) - associated teaching strategies

3. LB Platform

    - information needed to place the student (in this case with respect to this particular instructional goal)

- information needed to match the student with the instructional topic and teaching strategy

Components

As explained in Section 11, "Functional Organization" on page 73, Platforms are made up of components. It was also mentioned in that section that enhancements to the architecture of the system are possible by adding components and sub-components. This is done as outlined below.

- CSE Platform
  - new *domains* become new components
  - new *tasks* within a *domain* become new sub-components
- ACAI Platform
  - adding new *topics* become new (Instructional) sub-components
  - adding new *strategies* become new (Modulation) sub-components
- LB Platform
  - new *databases* become new components
  - new kinds of *database records* or groups of records become new sub-components

Section 20. CSE Level Description

Components - Domains - Tasks

Knowledge and information are organized (and/or used) within a set of AI tasks corresponding to the domains of the CSE. The components of the CSE have been established for the current model as knowledge domains. Additional tasks can be created as sub-components. At the established component level these are outlines.

Component - Controller

Domain: Maintain process control.

Task: Start and coordinate the services of the other components.

Component - Counsellor

Domain: Suggest an instructional path.

Task: The object is to understand the information contained in the LB Learners database, using it to asses the student's educational interests, needs and current standing. Effectively the revision model of the student is analyzed.

This picture is compared to its closest image in the approachable model. The goal is to create a progressed picture of the student, based on accepted developmental standards (what should this student be taught next). The Counsellor then suggests a path to meet the situation.

Component - Evaluator

Domain: Select an instructional path.

Task: Based on the suggested path the Evaluator identifies learning activities that match the goals set forth by the Counsellor. Information in the LB Instructionals database is used to select such activities. The goal here is to look ahead at the expected effect of a certain instructional activity (module), that is, to look at the image that this revision will create, and compare this to the next stage in the approachable model. This effectively computes the minimum closest augmentation factor for the revision model. The instructional is identified and the combined goals of both the Counsellor and the Evaluator are propagated to the next stage as met subgoals.

Component - Specialist

Domain: Modulate the instructional path.

Task: The information in the LB Modulations database is used as an instrument to asses the presentation approach from those available to the particular instructional selected. If no preference is indicate the standard sequence of strategies for teaching the topic is employed.

Component - Frameworker

Domain: Provide instructional services.

Goal: The way in which subgoals were previously met determine the goals at this step (as was also true before). The task, by default, is to deliver the activity in such a way that all previous subgoals are carried out. Opportunities for further tuning are limited at this point, since it is known what needs to be presented and how. Other less complex possibilities are available, however. One example may be intensity of presentation. This could come down from the Specialist as a limit on the amount of tries, or amount of time a student should spend on items in the activity.

Component - Monitor

Domain: Examine conditions periodically.

Task: While the Frameworker remains involved with the student during the course of the activity it does not monitor overall progress, but rather item by item progress. It is useful to keep separate tabs on how the activity is progressing and how the learner is reacting. While the Frameworker has as one of its goals to carry out the activity to completion, the Monitor has as its goal not to allow the student to loose interest or stop making progress. the Frameworker continues to carry out the activity to completion. But it is possible for this goal to become useless or to have exhausted its usefulness. In either case the situation is identified (one way would be by the hit/miss ratio of answers, but other criteria is likely depending on the activity at hand) and the Monitor may be triggered into action. It could cause a change of topic, by flagging the LB Learner-base with a signal that specifically directs the system to restart CSE processes. Alternatively the Monitor could simply request some form of recess, to later pick up the same activity where it was left off. This sort of "break" would have available modules in the ACAI to back it up. The Frameworker would have direct access to such modules and be able to present them with no further CSE intervention. These could be games or some other favorite activity.

Section 21. ACAI Level Description

Instructional topics, within subject areas are developed separately and become the sub-components of the ACAI. These are open-ended and therefore cannot be predetermined or established. Every time a new instructional topic is developed for E^3, one or more (typically many) Virtual ACAI components are created, depending on the possible combinations of the sub-components developed for it.

Sub-component - {ACAI(Instructional(Number Theory))}

For the chosen scenario three topics are presented. Each corresponds to a stage in the development of the topic. These can be loosely labelled as typical of grades Fourth through Sixth. However under E^3 there is no labelled difference. Any student who has the abilities (present in the current picture) necessary to gain access to this component (module) can escalate to any of its levels with measured activities in individual time frames.

Therefore all activities such as drills and exercises, and modulations for that topic should be grouped together in a single module. This also encourages independent development of topics within subject areas. While scope and sequence are logically selected by the system, the overall activity can overlap those definitions. This is consistent with how many topics are conventionally taught.

Some examples of what an instructional component for Number Theory could contain are given below.

Sub-component - (Instructional(Factoring))

Finding factors of numbers. This is a topic in Number Theory commonly taught to students in Fourth Grade.

Odd and Even Numbers: manipulative pairing of materials *(A number is defined as even if it can be divided by two with no remainder. A number is defined as odd if it has a remainder of one when divided by two).*

Sub-component - (Instructional(Common Factoring))

Finding common factors of two numbers. This topic is commonly taught in Fifth Grade.

Prime and Composite Numbers: *(A prime number is an integer that is evenly divisible only by itself and one; e.g., only itself and one are factors of the number). (A composite number is a number that has more than itself and one as factors).*

Sub-component - (Instructional(Greatest Common Factors))

Defining the greatest common factors of two numbers This topic is commonly taught in Sixth Grade.

Divisibility Rules: *(One number is said to be divisible by another number if and only if the first number divides evenly into the second leaving no remainder). (A number is divisible by 4 if and only if the last 2 digits of the number are divisible by 4) (A number is divisible by 8 if and only if the last three digits of the number are divisible by 8)*

Sub-component - {ACAI(Modulation(Number Theory))}

Typically the steps in teaching and learning this topic range from the very tangible (see/say) to the more abstract (expressing the concept). It is reasonable and within sound practice to present the topic in that sequence. However, it is entirely possible under E^3 to vary the sequence postponing or even bypassing those approaches that have been known not to work for the learner. This flexibility is relatively useful depending on the adaptability of the topic and the instructional methodology applied in the design of the Instructional. Some general examples of how this particular topic could be modulated are given.

Sub-component - (Modulation(visual recognition))

Odd and Even Numbers

Pictorial representations of many numbers

(boxes, circles, etc.)

Prime and Composite Numbers

Use of color codes to see patterns

in the Sieve of Eratosthenes:

Primes	Colors
five	= purple
seven	= blue
eleven	= red
all others	= black

1	2	3	4	5	6	7	8	9	10
11	12	13	14	15	16	17	18	19	20
21	22	23	24	25	26	27	28	29	30
31	32	33	34	35	36	37	38	39	40
41	42	43	44	45	46	47	48	49	50

Divisibility Rules

Pattern recognition by visual examination of examples:

(7891302 / 8, 3520038 / 4

any number ending with an even number

in the ones place will be divisible by 2)

Sub-component - (Modulation(auditory recognition))

Odd and Even Numbers

(Even + Even + Even = ?)

(Even + Odd + Odd = ?)

(Odd + Odd + Odd = ?)

Prime and Composite Numbers:

Based on the Sieve of Eratosthenes:

How many of the first 50 numbers are prime ?

Of the first 25 numbers, what percentage were multiples of three numbers?

Of the second 25 numbers, what percentage were multiples of three numbers ?

Divisibility Rules

10^1 / 4 does it divide evenly ? NO

10^2 / 4 does it divide evenly ? YES

Sub-component - (Modulation(abstract understanding))

Odd and Even Numbers

Do you know the rule ?

Do you need to test the rule for (Odd + Even + Odd = ?) ?

What other combinations do you know already,
without the need to test them ?

Prime and Composite Numbers

Based on the Sieve of Eratosthenes:

When is the first time that a multiple of 7 has not
been crossed out by a multiple of a smaller prime ?

What do you notice about multiples of 11 ?

Predict at which number the multiple of 11 will be crossed
out for the first time with no smaller primes as its multiple ?

Generalize a rule for what you discovered after answering
the last question.

Divisibility Rules

197836 / 4 divisible by 4 and 8 ?

YES Because 36 is divisible by 4

765872 / 8 divisible by 4 and 8 ?

YES Because 872 is divisible by 8

Note: The source of information on mathematics curricula was *Mathematics Methods for the Elementary and Middle School* (Bitter, 89).

Section 22. LB Level Description

Using the terminology introduced in Section 12, "Software Organization" on page 85, the various databases are considered components of the LB, and database records or groups of records are considered sub-components.

Component - {LB(Learners(Student1))}

For instance LB(Learners) could actually be the following set:

```
{LB(Learners(learner information),
            (observed mastery evaluations),
            (generated mastery evaluations),
            (mastery evaluation values))}
```

Components and sub-components can be added as necessary. The set above is illustrative not specific of the information that would be contained.

More specific information supplied to the above set could be obtained from standardized assessment test. An example is given below.

```
{LB(Student1(KeyMath.area performance.content),
            (KeyMath.area performance.operation),
            (KeyMath.area performance.applications),
            (KeyMath.subset performance.numeration),
            (KeyMath.subset performance.multiplication))}
```

For this particular student in the illustration, there is information available on test results of the *Key Math* standardized mathematics skills tests (Connelly, Nachtman, and Pritchett, 76). Each KeyMath LB sub-component would then contain test scores and their possible meanings. An LB Learners Component for student1 would then be specified as follows.

```
{LB(Student1(KeyMath.subset performance.numeration
            (10,average)),
            (KeyMath.subset performance.multiplication
            (14,superior)))}
```

The relevance of the information is linked to knowledge in the CSE if it uses it, and with how this information is mapped to that knowledge when in use. This will be seen in Section 24, "Inference Description" on page 168.

In addition to these forms of standardized data, It is suggested that information of a different nature could be kept, such as evaluations that are "observed" (observed mastery evaluations), and recorded in an ongoing basis, based on performance (during drills and exercises). This is simply a way of keeping score on performance.

It can be made substantially more complex than that however. A detailed description of computer-based test scoring measurements, leading to the development of learning prescriptions has been proposed (Bruno, 87), suggesting that such evaluations can be used to maintain a student information system.

But beyond that, it is also suggested that these kinds of information flows could be combined by the system to produce "generated" (generated mastery evaluations). This means that the system could combine gathered scores with expected performance scores (perhaps based on the KeyMath) and infer on the relative meaning of the assessment evaluation. Then use that information for further modulations, by adding new entries to some other appropriate subcomponent such as the "mastery evaluation values". Modifying those values in an ongoing basis the systems tunes itself to the current needs and progress of the student. While the exact parameters of the information cannot be outlined without a more detailed study, and would depend strongly on implementation decisions, the opportunity exists within the organization of E^3 to exploit information and gather information that is potentially very useful.

Specific storage format of the records is also subject to implementation specifications, as are the selection of LB records in terms of components and subcomponents. Those given here are conceptual illustrations.

Component - {LB(Instructionals(Number Theory))}

In the same manner the instructional software can be assessed as the student is being assessed. And thus for the Instructionals and Modulations subcomponents the sets that follow are possibilities.

```
{LB(Instructional(instructional component information),
                 (estimated effectiveness evaluations),
                 (generated effectiveness evaluations),
                 (effectiveness evaluation values))}
```

Component - {LB(Modulations(Number Theory))}

```
{LB(Modulation(component information),
              (anticipated usage evaluations),
              (generated usage evaluations),
              (effectiveness evaluation values))}
```

In both cases the goal is to rate the effectiveness of the software based on usage. Tentatively some initial rating is recorded as suggested by the "anticipated usage evaluations" and other ratings are recorded in an ongoing basis for each particular component when in use.

Consider for instance the following is an ACAI component.

```
{ACAI(Number Theory(odd and even numbers &
                    visual recognition))}
```

It resulted from the following combination.

```
{ACAI(Number Theory) <= Instructional(odd and even numbers) &
                        Modulation(visual recognition)}
```

"Estimated effectiveness" and "anticipated usage" would be initially recorded in format such as in the following example.

```
{LB(Instructional(odd and even numbers)
     (estimated effectiveness evaluations('value','range'))))}

{LB(Modulation(visual recognition)
     (anticipated usage evaluations('value','range'))))}
```

While their effectiveness when combined would be recorded separately, as in the following example.

```
{LB(Number Theory(odd and even numbers &
                  visual recognition)
                 (generated effectiveness evaluations
                  ('value','range'))))}

{LB(Number Theory(odd and even numbers &
                  visual recognition)
                 (generated usage evaluations
                  ('value','range'))))}
```

Value and range are substituted for some appropriate measurement criteria. In this manner various forms of assessment are possible and the effectiveness of the software can be understood more fully, even beyond its original purpose and design objectives.

Component - {LB(Inventory(Topics))}

This components is used by the system to locate specific available instructional paths. Information that could aid in that task includes:

- existing educational/instructional goals,
- existing educational/instructional sub-goals,
- generated educational/instructional requirements.

These would be sets of records that effectively inventory what is available to the system by component and sub-component, or more specifically by topic, as well as information on when these are applicable (prerequisites for applications). In this scenario for instance, Number Theory and its topics would be inventory as well as the fact that Multiplication as an instructional topic is a requirement for escalating to Number Theory.

In Section 24, "Inference Description" on page 168, it will seen that specific *patterns* in the knowledge-base question availability of specific kinds software. An inventory of all available software is then useful to map information in such a way that it can be more clearly seen how why these patterns are used. This component is a general facility but not a requirement to the structure of the system.

Section 23. Software Adaptability

Software Combinations

If the topic (Number Theory) as described earlier, was fully developed for this environment, there would be a number of sub-components available for possible adaptive combinations. Using the terminology introduced in Section 12, "Software Organization" on page 85, Instructional(Number Theory) would actually be a set such as the following.

```
Instructional(Number Theory(
                common multiples,
                understand multiples,
                odd and even numbers,
                find patterns in number sequences,
                greatest common factor,
                find least common multiple,
                prime and composite numbers,
                use factor trees,
                prime and composite numbers,
                prime factorization,
                divisibility rules,
                least common multiple,
                least common denominator)
```

Corresponding modulations for those topics are the following.

```
Modulation(Number Theory(
                visual recognition,
                auditory recognition,
                abstract understanding)
```

Remembering that conventionally these are treated as topics of a topic that expands at least three grade levels as outlined previously.

- Fourth Grade
  - common multiples
  - understand multiples
  - odd and even numbers
- Fifth Grade
  - find patterns in number sequences
  - greatest common factor
  - find least common multiple
  - prime and composite numbers
  - use factor trees
- Sixth Grade
  - prime and composite numbers
  - prime factorization
  - divisibility rules
  - least common multiple
  - least common denominator

A few of the possible combinations depending on individual cases would be the following.

```
{ACAI(Number Theory) <= Instructional(odd and even numbers) &
                        Modulation(visual recognition)}

{ACAI(Number Theory) <= Instructional(greatest common factor) &
                        Modulation(visual recognition)}

{ACAI(Number Theory) <= Instructional(prime factorization) &
                        Modulation(visual recognition)}
```

These possible combinations are, as has been described earlier, virtual components. No given ACAI component actual exists but are created in the above fashion. At given instance the above virtual components would become the following real components.

```
{ACAI(Number Theory(odd and even numbers &
                    visual recognition))}

{ACAI(Number Theory(greatest common factor &
                   visual recognition))}

{ACAI(Number Theory(prime factorization &
                    visual recognition))}
```

The set of all combinations for this component (number Theory) depends on the number of its I and M sub-components. The ACAI set for a given E^3 would correspond to the Approachable Model in that E^3. The way in which these combinations are applied, effectively the way in which the AM is applied, depends on the students and sound educational practice reflected in the CSE. A student may be proficient in this topic and capable of advancing to topics that are above his grade level. The student could master the entire topic regardless of grade level.

If the student however, has mastered all the topics in his grade level but cannot make appreciable progress in topics above his level these would be postponed (no further attempt would be made to present those topics). In that case escalation would be to a different topic (for example Divisibility) in his grade level with the corresponding selection of topics.

In cases where a particular modulation may help the student move further along within a topic, that avenue would be available for exploration. In short all of the things that teachers wish they could experiment with, to help students move at their own pace are possible here, without holding anyone else back, or boring any other student in the process.

Clearly teaching of topics in other subject areas would proceed in the same fashion, allowing the student to move as fast or as slow as he can in each area. This is not a new way of teaching, it is simply a way of using technology that facilitates opportunities for more individualized teaching programs.

Note: The source of information on KeyMath testing was *Assessing the Abilities and Instructional Needs of Students* (Hammill, 87).

Section 24. Inference Description

The descriptions given so far were meant to *illustrate* how curricula and curricular goals could map to E^3 components. But the key functions of the system depend on the components that support the knowledge-base, and these merit further description.

For a general description of Knowledge-based Systems refer to Section 2, "Necessary Information" on page 5. Additionally all AI terms used in this document are defined in the Glossary.

Mapping Information to Classes

For manipulating the information available in the LB within an inferencing environment it is necessary to create *templates* of this information in *working-memory*. This is done by designing corresponding *class member*. For instance the following information on a student:

```
{LB(Student1(KeyMath.subset performance.numeration
            (10,average)),
            (KeyMath.subset performance.multiplication
            (14,superior)))}
```

could be represented in working memory with corresponding attributes as follows.

```
Class Learner
      name(student1)
      KeyMath.subset performance.numeration
      (10,average)
      KeyMath.subset performance.multiplication
      (14,superior)
```

The preferred view for this scheme is that the LB component contain a sub-component for "student1" with information on KeyMath assessment tests. An implementation may prefer to store only information for which there are values. For example "14" in the KeyMath multiplication test. Therefore that record is stored in the LB. Information for which there are no values need not be stored, even when its corresponding attribute exists in the knowledge-base as part of a member of a class. The score, "14" in this case is considered superior and that interpretation could be recorded (this is also an implementation decision).

Class members would have to be defined that can support the same information in the form of attributes. Depending on the implementation, a comprehensive set of attributes would be defined upfront and more added when necessary. So for instance the starter set of attributes for the class Learner would possibly contain more than those attributes for which there are values in the LB, and would include all possible attributes for KeyMath. In that case that class would be defined as follows.

```
Class Learner
      name(student1)
      KeyMath.area performance.content
      KeyMath.area performance.operation
      KeyMath.area performance.applications
      KeyMath.subset performance.numeration
      (10,average)
      KeyMath.subset performance.multiplication
      (14,superior)
```

Clearly at least those attributes whose values are tested on the *LHS* of a *rule* must be defined (but not necessarily contain values) for the software to work.

Other LB components would be similarly mapped. The following is an example.

```
{LB(Modulation(Number Theory
               (mode(visual recognition)
               (usage(general & visual)
               (effectiveness(high))}

Class Modulation
      mode(visual recognition)
      usage(general)
      effectiveness(high)
```

In English, this LB component description in the CSE reads as follows.

> There exists a modulations component in the LB Platform associated with the Instructional Component Number Theory. For this component there exists a modulation called visual recognition which can be used in general (with most students) an in particular for visual learning. The (known/recorded) effectiveness of this modulation is high.

And the Class Member mapping, a little more concisely, reads as follows.

There exists a member of the Class Modulation
of mode=visual recognition, for usage=general
with effectiveness=high.

Encoding Knowledge

Knowledge encoded in the form of situation-action pairs, is a convenient form of representation for this application. However, while this model is presented as an AI knowledge-based system, there is no strong dependence on that paradigm for implementation. The rule-based approach is useful to the organization of this system and is therefore the premise for this description. But, it is in the architecture of this system where the approach rests, and the solution is founded on having broken down the problem into an effective system architecture.

With that in mind the following examples illustrate how the various pieces of the model work together, not of how they must be implemented.

Situation Description

Consider that Student1 has completed within some acceptable degree of performance the Multiplication Component of the ACAI under E^3. Multiplication as an instructional topic usually precedes Number Theory. Also consider that this student is in grade level four (Fourth Grade). The situation could be described as follows.

English Language Description

There is a student, named student1, of average intelligence and performance. Student1 is able to master most instructional materials for grade level four. This student does not particularly like or dislike mathematics.

In that area (Mathematics) the student has shown acceptable performance up to and including the topic of Multiplication. The topic that typically is taught following Multiplication is Number Theory.

For this topic (Number Theory) there are a number of concepts that could be taught to student1. Many of these concepts correspond to the grade level of the student. For these concepts a number of instructional strategies can be employed, all of which would be suitable in this case.

English Language Rules

This case description can be translated into rules. The English Language version of one such rule would be represented in fashion similar to the following example.

```
When: there is an average student in grade four
      who has completed multiplication

      and  there is a instructional software that
           follows multiplication (Number Theory)

      and  for this software there is a strategy compatible with
           the student's learning preference

Then: present instructional material to this student for
      the topic Number Theory using that strategy
      (visual recognition)
```

For a general purpose CMI/CAI System the above English Language rule would suffice and could be translated (written) in some rule based AI language. In E^3 the process is broken further broken down.

Rules by Component

The first *pattern* in the above general case rule:

```
there is an average student in grade four
who has completed multiplication
```

is knowledge relevant to the Counsellor's functions. That knowledge would be expressed differently and would cause other information to be created as follows.

```
When: there is an average student in grade four
      who has completed multiplication

Then: say in Student1 Component that this student
      is ready and capable to begin Number Theory,
      using any given entry strategy for that topic.

   ( Result: Temporary student1 data is created to reflect
             this potential interface )
```

The second pattern would be relevant to the Evaluator's function.

```
When: there is a instructional software that
      follows multiplication (Number Theory)

Then: ready this software at an entry level topic

  ( Result: Odd and Even Numbers is made ready
            Temporary Student1 data now reflects this )
```

Finally the third patterns triggers the creation of an ACAI component.

```
When: there is a strategy for Number Theory compatible
      with the student's learning preference and suitable
      for this topic

Then: ready this strategy for presentation

  ( Result: "visual recognition" is made ready
            Temporary Student1 data now reflects this )
```

The simple example above could certainly be written as a single rule, and not a very complicated one at that. However the point being illustrated is that of breaking down functions so as to yield information that can be isolated and stored. With this scheme a deeper model of the student begins to take shape.

Another reason is of course to be able to get to the level of granularity necessary for dealing with more complex problems while still being able to isolate cause and effect functions.

The organization leads to reduction of the various subproblems to its most fundamental aspects, thereby facilitating the kinds of substitutions that help to generalize to a set of widely used rules. Making the CSE generic is possible in this functional breakdown scheme by employing substitutions of the following representative categories:

- prerequisite,
- educational goal,
- educational tool,
- modulation, etc.

But more careful attention must be given to that possibility, in terms of both research and development. Work applicable to defining a correct set of substitution parameters has been started for similar reasons: knowledge acquisition improvement based on instructional variables associated with instructional strategies (Christenses and Tennyson, 87), and include:

- worked examples,
- amount of information,
- sequence of information,

- format of information,
- learning time,
- corrective error analysis,
- mixed initiative,
- advisement, and
- embedded refreshment and remediation.

The rest of the Process

The above example illustrates what is seen in Section 14, "CSE Process Scheme" on page 97 as one output of the Consideration Set. Since a scenario was chosen that places the student at the entry level of a topic, only one consequence is possible, the creation of an ACAI component such that:

```
{ACAI(Number Theory) <= Instructional(odd and even numbers) &
                        Modulation(visual recognition)}
```

now exits as:

```
{ACAI(Number Theory(odd and even numbers &
                    visual recognition))}
```

for the duration of the activity.

In more complex cases combination that exploit the facilities of ACAI processing would be possible and would be exercised as discussed in Section 15, "ACAI Process Scheme" on page 110. In this example the ACAI component (above) is passed to the Frameworker for presentation. Its work can be done with rules but need not.

The Monitor however could be a separate ruled-based program (subprogram or rule block) that is activated at this point by activating specific rules, making then available for firing. As data is collected during the session, a rule could then fire causing an interrupt, permanent (terminating the interface), or temporary (requesting a new ACAI for example).

An example such a (in its English version) is given below.

```
When: there is an ACAI active

      and the student has mastered the concept
      (successfully completed enough exercises)

Then: attempt to move on to a different ACAI
```

In this example the criteria applied (successfully completed enough exercises) determines that a new topic will probably be invoked. Then the entire E^3 process will be adjusted (recycled) as seen in Figure 4 on page 100.

If the criteria happens to be different, for example requiring that the student input his preference, it may become necessary to generate more exercises for the

current topic (the student may want to stay with it longer). In that case the Frameworker utilizing whatever is available to the current ACAI Component will attempt to sustain the activity without further CSE intervention.

If this is not the case and in fact a new ACAI is required, it may be produced from the current contents of the Consideration Set. If no suitable ACAI virtually exits in the Consideration Set, then a new inference cycle is invoke to request a new conflict set.

Note: The source of information knowledge-based systems was *Expert Systems in Data Processing* (Hellerstein, Klein and Milliken, 89).

Section 25. Example of a Model

Using the example a reduced version of the student model can be constructed. The student model exists implicitly. In this case its current complete picture contains some discrete pictures possibly in the area of Multiplication. For the purpose of this example these are not specified. But it is known that from Number Theory the concept of odd and even numbers has been a goal. Furthermore the modulation used to master this concept is also known.

During the next encounter with the system a Delta Model is formulated. This is an increase delta since it attempts to bring an image that is not currently in the Revision Model. The entire student model or the Living Model for Student1 is all of the concepts that can be learned (Approachable Model) the concepts that have already been learned (Revision Model) and at this moment the best possible change (Delta Model).

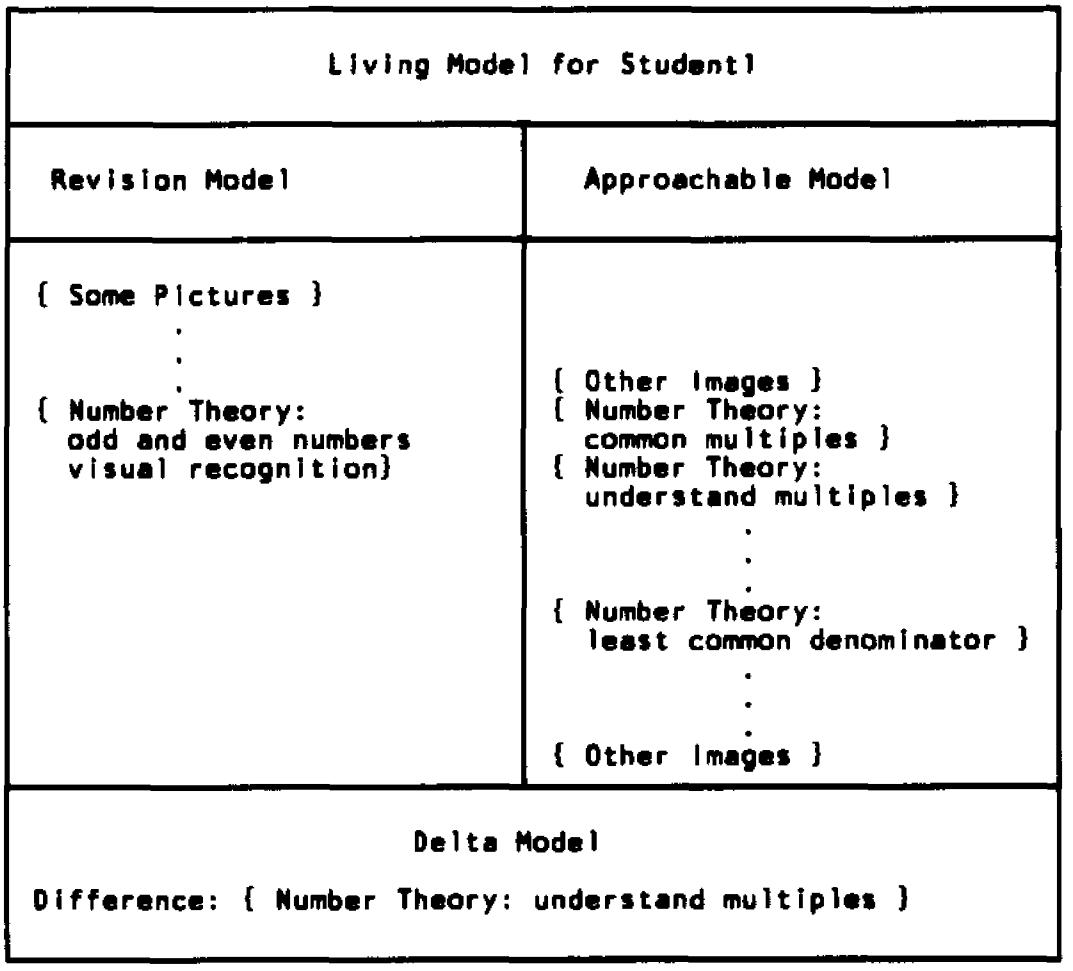

Figure 11. Example of a Student Model

Part Six: Summary

Section 26. Some Concluding Remarks

Deciding on the Approach

Since the E^3 Model has been presented as an application of Knowledge-Based Systems, some of the issues and discussion throughout this document take that premise. Selecting a development approach facilitates the exposition of the solution. For this reason one was chosen.

However since the solution is founded on an architecture that is independent of the implementation or knowledge representation approach it is not outside the scope of this research to consider other possibilities, such as logic programming or object oriented programming. Binding the E^3 solution to to paradigms in those approaches can be made as the subject of further research.

Ordering Knowledge

Within the knowledge-based paradigms it is important to emphasize that the underlying factors relating to the utilization of specific areas of knowledge cannot be arbitrarily tied to pieces of software. for this application. It has not been clearly established in the literature which piece of knowledge is applicable and useful in dealing with a given circumstance. But, rather it is the case that experience and intuition play an important role in the decision making process for an educator. For that reason overall processes are tied to components and these are involved by proving, testing and gathering of data.

The use of priority rules, sometimes common to this type of knowledge-based approach, and which exist as tools in several AI languages, including OPS5 and KnowledgeTool, is not advisable here. Grouping of knowledge inside groups of knowledge by priorities is useless for this application, since it is not known until diagnosis and evaluation have taken place the desired sequence of knowledge application. These activities are in themselves part of the overall process of the CSE and no priorities can be exercised in general.

Furthermore, this estimated inability to prioritize is not a deficiency in design but rather a strength. Priorities, and priority rules have been viewed by some as a handicapping factors and objectionable flows, concerning the difficulty involved in integrating new heuristics with existing ones. Even in the presence of priorities the problem is often addressed as a labor intensive programming task, that is, the knowledge engineer must hack the rules to achieve the new desired results, in an effort sometimes referred to as refinement. In addition there is sometimes obvious misuse of the rule-based paradigm by forcing flows of logic with the use of priorities.

This model depends on fine decomposition of knowledge coupled with a well defined classification of information to achieve the desired results. The ability to extensively classify information is of importance, particularly here, because the relative importance of each informational attribute, in each case study, or for each IEP, seem to depend largely on factors outside the CSE model. Informational factors which emerge (in this case from the LB component) are more

determining than any paradigm that could be employed. This formalism was seen in Davis (84). In device-centered models of physical systems each physical device maps directly into a structured object in the representation. This type of representation is more complex, because a more complex control structure is required to support it. In E^3 such an independent control structure has been added to the CSE process (the Controller) to account for some of the requirements inherent to this scheme.

Section 27. Preliminary Implementation Suggestions

The Mathematics scenario presented earlier provides some clues as to how a system such as this one could be implemented. While not outlined as requirements, part of the feasibility of E^3, both financial and technical, depends on the tools employed. The question asked here is how can the technical issues be conquered without exhausting the bounds of available and reasonable means.

Considerations spread across all the areas involved. On the Computer Science side the concern is hardware and software possibilities and constraints; on the Education side, technical expertise and cost. The underlying premise is that not only must the system be good, but also achievable, thus usable.

Knowledge Representation Language

While more than one avenue for implementation is possible, at least one is outlined here. A vehicle for future implementation of E^3 is KnowledgeTool, a knowledge-based programming language. It has been seen, in HARPY (*The Handbook of AI*, Barr and Feigenbaum, 81) for instance, that diverse forms of knowledge can be integrated into a single precompiled network. That technique is inherent to knowledge-based systems implemented in KnowledgeTool.

In addition a feature of the language that allows for *procedural matches* lends itself well to ACAI processing, which will benefit from the combination of rule-

based and procedural programming paradigms. Also its ability to integrate with more conventional programming tools, such as a variety of database management systems, facilitates the implementation of a deep model.

KnowledgeTool was designed and developed at T.J. Watson Research (Cruise *et al.*, 86) to tackle very large problems and thus offers several advantages consistent with E^3 requirements. In general KnowledgeTool provides a flexible approach to implementing complex applications, but more specifically for E^3 it offers the advantages outlined below.

1. KnowledgeTool combines procedural and ruled-based programming, providing the full power of both programming paradigms. This is a necessary requirement for the ACAI scheme.

2. It is highly effective for rapid prototyping.

3. Its use of knowledge oriented rule subroutines seems naturally suited to the kinds of knowledge areas occurring in the subdomains of computer-based education. Partitioning into groups of rules is essential to CSE processes that depend on isolation of functionally related pieces of knowledge.

4. It supports complex representation of knowledge that express complex relationships, both within conditions and actions.

5. It supports sophisticated operations on both the condition and the action sides. These are two powerful advantages where relatively unexplored and presumed complex applications are concerned. The problem of codifying

and capturing knowledge from scores of highly trained professionals in the area of Education and Psychology is as hard as the hardest problem attempted by knowledge-based systems.

6. It provides good runtime performance, since it was designed for a real-time application environment. This is essential to the on-the-fly production of IEP(s).

7. It provides extensive programming control over inferencing, needed for such a complex application. This facility was in part the basis for the Consideration Set approach.

8. Its data-driven form of inference is appropriate for an activity largely dependent on information used in decision making. This is an appropriate approach for the implementation of the learner model.

9. It provides good access to other kinds of existing software and interfaces. This is convenient for instructional software proliferation.

The Mid-Range Computer

On the hardware side, the concept of host serving machine coupled with a number of workstations seems most adequate for the realization of this system. That general philosophy appears sound as was originally demonstrated with PLATO. More recently the Witcat System 300, the Houghton Mifflin Dolphin Curriculum and the TOAM Computer Aided Instruction System, have been

reported to employ such a philosophy. (Balajthy, 88). And, the system attributed with being the most financially successful in the market, MICROHOST by Computer Curriculum Corporation (Bork, 85) is also an example of this approach. In general terms the report as expressed by Balajthy is as follows.

> A minimum of teacher training is necessary to use Integrated Learning Systems (ILS), less than is required for use of the microcomputers. Supervision responsibilities appear minimal, as well. Children seem to be able to function independently.

The value of microcomputing in this field is not underestimated and thus it is suggested that the same operational concept could be applied employing one or more micros serving hosts and several others used as workstations. In either case, it is process distribution that must be maintained for the purposes presented in the architecture.

Ideally then, a mid-range system, that is both generally more servicing than microcomputers and more cost effective than large scale computing should be employed. For that purpose the IBM AS/400 machine provides a good fit. In addition it also provides an environment that is operationally simple (to a relative degree) while still supporting KnowledgeTool and the complexity of applications that merit that kind of implementation.

Implementation Recommendations

Based on the formulated functional specifications for E^3 and the above arguments, preliminary implementation recommendations seem appropriate. This is

done in an effort to encourage appropriate implementation if one is taken up independently of this research.

While the general ideas in E^3 can be adapted to other operational environments, these suggestion seem the most logical, since the design of E^3 was based on understanding of the relevant implementation factors as they relate to the specific of the tools in mind. A prototype implementation should approximate the guidelines given below.

- E^3 requires a development environment at least as powerful and flexible as that provided by KnowledgeTool for the CSE Platform.

- For the ACAI Platform a variety of programming languages are perfectly suitable including BASIC, COBOL, PASCAL or C.

- The preferred operational environment is the AS/400 System, using PS/2 computers as workstations.

- Future research extensions should include services that better exploit features of workstations in distributed environments.

  This would enhance the potential value of E^3 for the integration of education among the home, classroom and tutor. It would make it possible to take home software that has been individually prepared for a learner, by the hosting E^3 at an educational facility and use it at home in a home computer workstation, or in transit with a portable device.

- In addition, other features such as the *Audio-Visual Connection (AVC)* (an enhancement facility for knowledge representation languages and KnowledgeTool applications that utilize PS/2 workstations in particular, capable of producing special visual and sound effects) will add the potential to meet more sophisticated educational goals, such as are required in special education.

Reaching the Classrooms

E^3 was designed with the goal in mind to take advantage of the best mid-range computers have to offer in terms of high power and performance for high affordability. At the same time the design is constrained in such as way that low end implementations are possible, in the server to workstations philosophy. However, it is the case that this system is also a natural for the large systems environment as well. As social and environmental pressures modify the ways in which the education of our children is provided for, alternative educational settings will have to be the answer to the more pressing problems.

Witnessing the revival of the one room school-house, this time set not only among the trees, but also inside office buildings, the rationality of such a solution is not generally being questioned, but rather its feasibility and acceptance by parents and educators. Where the educational army of people needed to meet the needs of our children steadily declines and cannot exists in certain settings, technology again must come in and lend a helping hand.

It has been suggested and already tried by educational leaders such as Dr. Fernandez (New York Times 3/90), that classrooms be opened in the business setting for the children of employees, to alleviate day-care and other social problems. But obvious question are then asked.

- What kinds of facilities can be provided for those students?
- Without the enormous wheels of the school setting in motion the following questions are pertinent.
  - How can even the simplest special service be provided for?
  - What number of students and what grades will be serviced in those classrooms?
  - How many classrooms?
  - How many teachers?
  - How often will the above parameters change during course of attendance for each student?
- In what ways should technology be employed to facilitate this effort?

One common resource many of those one-room school-houses will have available to them is the use of computers. A corporation that is interested enough in people and Education to house a classroom, would probably find ways to lend some CPU cycles to the cause. The value of the E^3 concept in this case need not be explained further.

Section 28. Contributions

Successful completion of E^3 would provide contributions to two historically distinct fields broadly identified as Computer Science and Education. While the proposed work may be said to lie at the crossroads of both, this document brings out issues that are important to both. And both have been serviced neutrally well.

From the perspective of Computer Science it comes at a time when the validity and reasonable value of AI techniques is being questioned. E^3 would demonstrate how with sobriety and intent of purpose a system that employs AI techniques can be made far superior to conventional computer-based systems.

For the Education community it conveys a message that advanced computer technology, when carefully selected and comprehensively applied may bring us closer to the utopia than to the next crisis in education. It also will clearly show that only via a truly interdisciplinary effort can any approach succeed.

Finally it is important to consider that perhaps only as a doctoral thesis, representative of a purely academic endeavor, can an honest common solution be found that satisfies both disciplines. The potential material value of computer-based education as an effective implementation of advanced systems technology to some degree prevents a real answer from coming out of any other sector.

Research Contributions

E^3 does several significant and original things.

- It combines:
  - inspiring established and conservative processes from special education with
  - knowledge-based computer technology.
- It introduces the concept of Computer Supported Education and provides a classification of functionality for such systems.
- It employs a highly adaptive modeling scheme with special definitions.
- It employs software combination techniques that are unique for an educational system, as a natural consequence of having arrived at a good design organization.
- It fosters the principles of:
  - fined-grained programming,
  - functional isolation,
  - deep modeling, and
  - software reuse.

These now are presented as how they relate to the two research areas.

Relating to Education

A general framework was presented for the design of computer-based educational systems that does the following.

- Introduces the concept of CSE as an educational system.
- Encourage the employment of a well defined integrative architecture for the design and development of CSE Systems.
- Proposes the approach of independently yet organized development of components and subcomponents.
- Proposes this approach as a generic framework independent of the instructional theories applied in the design of components and subcomponents.
- Formally incorporates the concept of Instructionals and Modulations as interactive sets.
- Provides an information oriented student model.
- Facilitates the extraction of student as well as instructional assessment data.
- Provides facilities that eliminate competition among educational paradigms, in exchange for coexistence.

- Encourages independent research both in instructional software design and the empirical aspects of educational research.

Relating to Computer Science

A general framework was presented for the design of computer-based educational systems that does the following.

- Addresses the design of modeling schemes for CSE with analogous applications to other areas of knowledge representation.
- Fuses procedural, rule-based, modeling and software reuse as part of the concerted design architecture.
- Brings computer-based education to higher theoretical standards.
- Encourages independent research of instructional software design tools.
- Introduces a heuristic model of instructional/learner dynamics.

Open Areas of Research

These are many areas for follow-up research. Some suggestions follow.

- A model of distributed inference, between the hosting environment and the servicing environment should be selected or proposed.

- A set of fundamental activities for the CSE Components must be newly proposed or extracted form the literature.

- A form of classification, or Taxonomy for computer-based instructional materials, must be used to guide the process of acquisition of new ACAI Components.

- Ways of exploiting existing data banks of student information should be considered.

- Other approaches should be explored for the architecture that include object-oriented programming, possibly defining the LB as an object-oriented database and the CSE as a set applicable Methods.

- Communication methodologies among active environments of this kind should be applied to tie in as an active educational network.

List of References

Books and Technical Reports

[Anderson 85] Anderson, J.R. *Cognitive Psychology and its Implications*, Freedman, New York, NY., 1985

[Andriole 85]Andriole, S.J. *Applications in Artificial Intelligence*, Petrocelli Books, NJ., 1985

[Balajthy 88] Balajthy, E. *Recent Trends in Minicomputer-Based Integrated Learning Systems for Language and Arts Instruction*, Dept. of Elementary and Secondary Education and Reading, State University of New York at Geneseo, Paper presented at the Rutgers University Spring Reading Conference, New Brunswick, NJ., 1988

[Barnett 82] Barnett, J.A. *Some Issues of Control in Expert Systems*, USC/Information Sciences Institute, Marina del Rey, CA., 1982

[Barr 81]Barr, A., Feigenbaum, E.A. *The Handbook of Artificial Intelligence*, Heuristic Press, CA., 1981

[Belsches-Simon 84] Belsches-Simon, G., Lines, P. *The Legal Rights of Handicapped Students*, ECS Footnotes, 1984

[Bigge 62] Bigge, M.L., Hunt, M.P. *Psychological Foundations of Education*, Harper and Row Publishers, NY., 1962

[Bishop 81] Bishop, T.D. *Application of Microcomputers to Existing Mathematics Courses*, American Mathematical Association Meeting of Two Year Colleges, New Orleans, LA., 1981

[Bork 85] Bork, A. *Personal Computers in Education*, Harper and Row, New York, NY., 1985

[Brudner 72] Brudner, H.J. *Technology in Education: Challenge and Change*, Charles A. Jones Publishing Company, Worthington, OH., 1972

[Bruno 87] Bruno, J.E. *Using Computers for Instructional Delivery and Diagnosis of Student Learning in Elementary Schools, Computers in the Schools*, The Haworth Press, New York, NY., 1987

[Bushnell 67] Bushnell, D.D. *The Computer in American Education*, John Wiley, NY., 1967

[Christenses 87] Christenses, D.L., Tennyson, R.D. *MAIS: An Empirically-Based Intelligent CBI*, Annual Convention of the Association for Educational Communication and Technology, Atlanta, GA., 1987

[Connelly 76] Connelly, A.J., Nachtman, W., Pritchett, E.M. *KeyMath Diagnostic Arithmetic Test*, American Guidance Service, Circle Pines, MN., 1976

[Cott 85] Cott, A. *Help for Your Learning Disabled Child, The Orthomolecular Treatment*, Times Books, NY., 1985

[Cruise 86] Cruise, A., Ennis, R., Hellerstein, J., Masullo, M.J., Milliken, K.R., Rosenbloom, M., VanWoerkom, H. *YES/LI A Language for Implementing Real-Time Expert Systems*, IBM Research Report, 1986

[Davis 84] Davis, R. *Diagnostic Reasoning Based Structure Behavior*, The Artificial Intelligence Laboratory, MIT, 19984

[Davis 82] Davis, R., Lenant, D.B. *Knowledge-Based Systems in Artificial Intelligence*, McGraw-Hill, NY., 1982

[Dick 65] Dick, W. "The development and Current Status of Computer-based Instruction", *American Educational Research Journal*, 1965

[Fischer 82] Fischer, G., Lemke, A.C., McCall, R. *Towards an Architecture Supporting Contextualized Learning*, University of Colorado, Boulder, CO., 1982

[Forgy 82] Forgy, C. "Rete: A Fast Algorithm for the Many Pattern/Many Object Pattern Match Problem", *Artificial Intelligence*, Vol. 19, No. 1, September 1982, pp. 17-37.

[Gadow 86], Gadow, K.D. *Children on Medication - Volume I, Hyperactivity, Learning Disabilities, and Mental Retardation*, College Hill Press, CA., 1986

[Griesmer 84] Griesmer, J.H., Hong, S.J., Karnaugh, M., Kastner, J.K., Schor, M.I., Ennis, R.L., Klein, D.A., Milliken, K.R., VanWoerkom, H.M., *YES/MVS: A Continuous Real Time Expert System*, IBM Research Report, 1984

[Haga 67] Haga, E. *Automated Educational Systems*, IL., 1967

[Hammill 87] Hammill, D.D. *Assessing the Abilities and Instructional Needs of Students*, Pro-ed, Austin, TX., 1987

[Hellerstein 89], Hellerstein, J.L, Klein, D.A., Milliken, K.R. *Expert Systems in Data Processing*, Addison-Wesley, CA., 1989

[Henderson 79] Henderson, R.A. "Economic Implications of Public Education of the Handicapped", *Journal of Research and Development in Education*, (p 71-79), 1979

[Hofmeister 86] Hofmeister, A., Ferrara, J. *AI Applications in Special Education*, Special Education Programs, Utah State University, June 1986.

[Hong 89] Hong, J-C. *The Applications of Expert Systems in CAI*, National Taiwan Normal University Computer Education, 1989

[Hotard 88] Hotard, S.R., Cortez, M.J. *Using Computer Assisted Instruction to Raise and Predict Achievement in Chapter I Students*, Education Consolidation Improvement Act Chapter 1, LA., 1988

[Hughes 77] Hughes, J.K., Michtom, J.I. *A Structured Approach to Programming*, Prentice Hall, NJ., 1977

[Jacobs 86] Jacobs, A. *Master Plan for Instructional Computing*, Arizona Office of Educational Development, 1986

[Lenzen 55] Lenzen, V.F. *Procedures of Empirical Science*, International Encyclopedia of Unified Science, 1955

[McGraw 88] McGraw, K., Brown, B. *Designing an Intelligent Curriculum Planner*, Texas Instruments Defense Systems and Electronics, Artificial Intelligence Laboratory, TX., 1988

[Masullo 88] Masullo, M.J., Sjolund, M. *Computers and Handicapped Children: AI and LEKOTEK's COMPUPLAY*, IBM Research Report, 1988

[Ennis 86] Cruise, A., Ennis, R.L., Finkel, A., Hellerstein, ., Loeb, D.J., Klein, D.A., Masullo, M.J., Milliken, K.R., VanWoerkom, H., Waite, N. *Isolating Functionality in a Real Time Expert System*, IBM Research Report, 1986

[Milliken 86] Milliken, K.R., Cruise, A., Ennis, R.L., Finkel, A.J., Hellerstein, J., Loeb, D.J., Klein, D.A., Masullo, M.J., VanWoerkom, H., Waite, N.B. *YES/MVS and the Automation of Operations for Large Computer Complexes*, IBM Research Report, 1986

[Olson 76] Olson, D., Magero, J. *Audio-Visual Instruction*, 1976

[Park 87] Park, O., Perez, R. and Seidel, R. *Intelligent CAI: Old Wine in New Bottles, or a New Vintage Artificial Intelligence and Instruction: Applications and Methods*, Addison-Wesley, CA., 1987

[Parry 85] Parry, J.D *Mandate Consultant: An Expert System for Examining the Implementation of Special Education Regulations*, Annual Meeting of the American Association of Mental Deficiency, Philadelphia, PA., 1985

[Petkovick 85] Petkovick, M.D., Tennyson, R.D. "A few more Thoughts on Clark's *Learning from Media*", *Educational Communication and Technology Journal*, 1985

[Petkovick 84] Petkovick, M.D., Tennyson, R.D. "Clark's Learning from Media: A Critique", *Educational Communication and Technology Journal*, 1984

[Pfeiffer 68] Pfeiffer *New Look at Education: Systems Analysis in Our Schools Colleges*, Odyssey Press, NY., 1968

[Rayner 72] Rayner, G.T. *An Empirical Study of a Methodology for the Revision of Systematically Designed Educational Materials*, Office of Naval Research, 1972

[Reigeluth 83] Reigeluth, C.M. *Instructional-Design Theories and Models: An Overview of their Current Status*, Lawrence Erlbaum Associates, Publishers, Hilldale, NJ, 1983

[Richter 81] Richter, S.J. *A Computer-Managed Tutorial Reading System*, Ph.D. Thesis, University of Alabama, 1981

[Roblyer 88] Roblyer, M.D., Castine, W.H. *Assessing the Impact of Computer-Based Instruction: A Review of Recent Research*, Haworth Press, New York, NY., 1988

[Salend 84] Salend, S.J., Zirkel, P.A. *Special Education Hearings: Prevailing Problems and Practical Proposals*, Education and Training of the Mentally Retarded, 1984

[Senese 83] Senese, D.J. *Educating for Excellence: The Role of Instructional Technology*, University of North Carolina, Wilmington, NC., 1983

[Senese 83] Senese, D.J. *Our Future Growth is Tied to Educational Technology*, Telecommunications Public Forum Marshall, MN., 1983

[Schank 87] Shank, R., Farrell, R. *Creativity in Education: A Standard for Computer-based Teaching*, Yale Artificial Intelligence Project, Dept. of Computer Science, Yale University, February 1987

[Schank 85] Shank, R., Slade, S. *Education and Computers: An AI Perspective*, Yale Artificial Intelligence Project, Dept. of Computer Science, Yale University, October 1985.

[Shortliffe 76] Shortliffe, E. *Computer-based Medical Consultation: MYCIN*, Elsevier, NY., 1976

[Silberman 69] Silberman, H.F. *Applications of Computers in Education*, Computer Assisted Instruction, Academic Press, NY., 1969

[Silver 84] Silver, L. *The Misunderstood Child*, McGraw Hill, NY., 1984

[Smith 81] Smith, T.E. "Status of Due Process Hearing" *Exceptional Children*, 1981

[Soulier 88] Soulier, J.S. *The Design and Development of Computer-based Instruction*, Allyn and Bacon, MA., 1988

[Splittgerber 79] Splittgerber, F.L. "Computer-based Instruction: A Revolution in the Making?", *Educational Technology*, 1979

[Stubbs 85] Stubbs, D.F., Webre, N.W. *Data Structures with Abstract Data Types and Pascal*, Brooks/Cole Publishing Company Monterey, CA., 1985

[Turnbull 81] Turnbull, A.P., Strickland, B., Turnbull, H.R. *Due Process Hearing Officers: Characteristics, Needs and Appointment Criteria*, Exceptional Children, 1981

[VanLehn 88] VanLehn, K. *Towards a Theory of Impasse-Driven Learning*, Learning Issues for Intelligent Tutoring Systems, 1988

[Vellutino 79] Vellutino, F.R. *Dyslexia: Theory and Research*, The MIT Press, MA., 1979

[Waterman 86] Waterman, D.A. *A Guide to Expert Systems*, Addison-Wesley, CA., 1986

[Watson 72] Watson, P.G. *Using the Computer in Education*, Educational Technology Publications, NJ., 1972

[Weinberger 71] Weinberger, G. *The Psychology of Computer Programming*, VanNostrand Reinhold Company, NY., 1971

[White 79] White, M.A. *The Future of Electronic Learning*, Lawrence Erlbaum Associates, NJ., 1983

[Winston 79] Winston, P.H. *Artificial Intelligence*, Addison-Wesley, CA., 1979

Informational Documents

AS/400 Application Development Tools: Programming Development Manager: User's Guide and Reference SC09-1169-00, IBM Corporation

A Guidebook for Committees on Special Education in New York State, The University of the State of New York, The State Education Department, Albany, NY., 1987

IBM THINK Magazine, Number 5/1988

Commission on Standards for School Mathematics of the National Council of Teachers of Mathematics, Curriculum and Evaluation Standards for School Mathematics, Reston, VA., 1987

KnowledgeTool Translator, User's Guide and Reference, SC21-9849-00, IBM Corporation

OPS5 User's Manual, L. Forgy,CMU-CS-81-135 Dept. of Computer Science, Carnegie-Mellon University, July 1981.

T.H.E. JOURNAL, Dec./Jan. 88/89

Vocabulary for Data Processing, Telecommunications and Office Systems, GC20-1699-6, IBM Corporation