The Evolution of Instructional Design Principles for Intelligent Computer-Assisted Instruction

Dr. Christopher Dede
School of Education
University of Houston—Clear Lake
and
Dr. Kathleen Swigger
Department of Computer Science
North Texas State University

The research reported herein was supported, in part, with funds from the Air Force Office of Scientific Research (AFOSR) and sponsored by the Air Force Human Resources Laboratory, Brooks AFB, Texas. Partial funding was also supplied by the University of Houston—Clear Lake.

Since the early use of computers for educational purposes, one approach to instructional applications, computer-assisted instruction (CAI), has been dominant. This paradigm is designed to optimize the teaching capabilities of the computer, given the constraints of slow processor speed, small internal memory, limited external storage, and expensive user access. Because these constraints have been universal during the first four decades of computer evolution—especially for schools with limited financial resources per student—until recently CAI has been the predominant methodology used for instructional computing.

In 1970, a research scientist in artificial intelligence, Dr. Jaime Carbonell, put forward an alternative paradigm for using the computer as teacher (Carbonell, 1970). His approach, termed intelligent computer-assisted instruction (ICAI), was designed to optimize the teaching capabilities of the computer based upon opposite assumptions about hardware: fast processors, large internal and external memories, and inexpensive availability to users. Initially, due to limited computer power, ICAI applications were created only by small groups of researchers at advanced facilities. Within the next five years, however, computers capable of fulfilling Carbonell's assumptions should be readily available to schools at costs within their instructional budget.

With the advent of powerful, inexpensive school computers, ICAI is emerging as a powerful rival to CAI. In response, both types of developers are making claims about the relative validity and importance of their respective methodologies. Some CAI advocates argue that ICAI is not fundamentally different from their own work, representing an extension of well-established CAI methods rather than a radical shift (Scandura, Stone, & Scandura, 1986). Other CAI proponents do see ICAI as revolutionary rather than evolutionary, but contend that well-constructed CAI courseware is as effective for many educational applications as ICAI programs—in addition to being cheaper, more robust, and more easily developed (Bork, 1986).

Some ICAI developers respond by describing frame-oriented, branching CAI materials as *intrinsically* inflexible, insensitive to students' needs, and ineffective compared with "intelligent" coaches and tutors who understand what, whom, and how they are teaching (Keller, 1987). Also, CAI courseware is seen as restricted in the complexity of the subject matter and cognitive goals that can be conveyed, due to the "combinatorial explosion" of branching frames that are needed to optimize preprogrammed sequences to the needs of a variety of learners. At present, almost all current CAI authoring systems are limited to this branched programmed instruction model (Merrill, 1985). However, the design of more sophisticated authoring systems based on ICAI principles is underway (Lewis, Milson & Anderson, 1986; Tennyson & Christensen, 1986).

The issue of instructional systems design (ISD) complicates the debate still further. ISD theorists argue that their models are valid for all types of instruction, whether conducted by human teacher, CAI courseware, or

With the advent of powerful, inexpensive school computers, ICAI is emerging as a powerful rival to CAI.

"Intelligent" tutors and coaches are neither like people ... nor like CAI devices.

ICAI device (Snelbecker, 1983). However, CAI developers tend to use somewhat different components of ISD theory than do human teachers, reflecting the divergent instructional attributes of computers and people (Halff, 1986). For example, ISD prescriptions are often seen as annoyingly precise by people (who use "common sense" to interpret general statements), but are insufficiently specific for formal application in an unintelligent computer program.

To the extent that AI-based devices are dissimilar from CAI courseware in their instructional characteristics, ICAI development may require other types of ISD principles. "Intelligent" tutors and coaches are neither like people, who have peripheral real-world knowledge, associational long-term memories, and limited short-term memory capacity, nor like CAI devices, which lack fuzzy logic and dynamic models. Current ISD approaches have been built around empirical experience with human and CAI instruction, so existing models may not be as generic as supposed, and new types of theoretical instructional design initiatives may be necessary (Duchastel, 1987).

For example, standard instructional design practice does not separate the production of course material from teaching style issues. The designer may use implicit instructional heuristics in developing the flow of content, but does not delineate a separate, explicit set of pedagogical, strategic, and tactical rules for implementing the course. In contrast, the modularization of pedagogy and content expertise in an ICAI system requires a design environment which articulates content and presentation independently. Design tools which reflect an ICAI philosophy of instruction are now under development (Russell, Moran & Jordan, 1986).

This article presents the following perspective on the issues described above:

1. ICAI and CAI are different paradigms for using the computer as teacher, based on dissimilar development methodologies and instructional design principles.
2. Existing ISD models may be insufficient for the evolution of artificial-intelligence-based educational devices. The instructional design theories most likely to be useful for ICAI are incompletely developed and may require new theoretical initiatives.

This viewpoint is contrary to "conventional wisdom" about ICAI and ISD, and this article is written in the hopes of stimulating a dialogue about these issues.

To limit the scope of the discussion, the instructional systems contrasted in this study are "active" rather than "passive." A continuum can be described between active, teacher-centered courseware, which has a structured instructional sequence and specific objectives against which learning will be evaluated, and passive, discovery experiences in which the learner is placed in an information-rich environment to be explored without guidance or targeted outcomes. An example of active CAI systems is tutorial courseware; the equivalent in ICAI are programs utilizing embedded intelligent tutors and coaches. (ICAI and CAI developers also build more discovery-oriented applications, but the differences between the two paradigms are most easily illustrated through contrasting active systems.)

CAI: Its Instructional Design and Development Methodology

First, it is important to note that there is no single instructional design methodology for developing CAI materials because there is no single type of CAI program. Most of the literature on CAI (e.g. Bork, 1981; Steinberg, 1984) enumerates a number of different types of instructional programs under the broad category of computer-assisted instruction. The list of different types of CAI applications includes games, simulations, problem solving, drill and practice, and tutorial programs.

One dominant perspective within this variety stems from the fact that most books about CAI emphasize design techniques for developing drill-and-practice or tutorial materials rather than games, simulations, or problem-solving applications. This is unfortunate because tutorial design techniques tend to be very similar to those used to

Existing models may not be as generic as supposed, and new types of theoretical instructional design initiatives may be necessary.

develop programmed texts and primarily stress the mastery of specific objectives as quickly and efficiently as possible. Such a view is highly related to an associational perspective on learning, such as that advocated by Skinner (1953).

According to this school of thought, neural networks are established by providing reinforcing stimuli each time the student answers correctly. Although Skinner is certainly not the father of CAI, behavioristic ideas have greatly influenced the programmed text and the CAI community. For example, most authors of CAI textbooks believe that the selection of content, the type of feedback, and other reinforcing stimuli used to maintain and regulate student effort must be carefully controlled. Through such a stimulus/response structure, the designer is seen as shaping more complex behaviors through building up student response chains composed of small steps. Critics of traditional CAI are quick to note that such programs are automatic "page turners" that condition students rather than teach them.

A second major influence on the development of CAI materials has been the use of authoring languages to assist designers with preparing these programs. Almost from the beginning, these authoring aids, such as IBM's COURSEWRITER, Plato's TUTOR, Hewlett-Packard's IDF, and PILOT, have incorporated design techniques which promote the development of the tutorial and drill-and-practice modes of CAI. Historically, authoring languages have forced designers to look at the instructional process as a series of frames composed of right/wrong answers and replies, unanticipated answers and replies, branches to and from specific remedial areas, and counters for correct and incorrect responses.

In such a framework, typically the designer selects a representative set of tasks and problems by considering their detailed step-by-step solutions. However, these analyses are often performed solely at a logical level rather than also considering the psychological processes that are involved. To a large extent, these limitations are imposed on the author by the authoring language itself, since responses must be anticipated and identified through prestored keyword matching schemes.

Critics of traditional CAI are quick to note that such programs are automatic "page turners" that condition students rather than teach them.

To make the program writing exercise even more facile, the question-answer sequences are structured to proceed in relatively small steps. As a result, the responses themselves tend to be highly structured, often in a framework of multiple choice or a restricted set of entries. Thus, many authoring aids become, in fact, implicit design methodologies for CAI materials.

The most widespread, scientifically based theories of design for CAI are based on the more general instructional design approaches, such as Gagne and Briggs (1977) or Merrill (1983). These theories build on a behavioristic base. For example, Briggs (1979) defines human learning in terms of procedures that might be appropriate to the design of instruction. This model proposes a methodology that includes techniques such as stating clear behavioral objectives, developing a pretest and a posttest, performing a task analysis, developing the instructional material, and performing formative/summative evaluations.

Similarly, Merrill's component display theory (CDT) postulates that, for each type of objective, a unique combination of primary and secondary presentation forms will most effectively promote student acquistion of that objective. Learning outcomes are classified on two dimensions: type of content and task level required of the student. Merrill's theory is instructionally oriented in that it presents criteria that determine the appropriate presentation modes for different types of learning. Both Brigg's and Merrill's theories emphasize (1) breaking down complex tasks into elementary motor operations ("learning hierarchies" derived from Gagne's cumulative learning theory) and (2) breaking down ambiguous instructional objectives (such as "understanding" or "appreciating") into observable and unambiguously accessible terminal behaviors that represent motor operations.

In CAI, the computer does not "understand" the content, student, or pedagogy involved....

A cognitive science approach to instructional design that has gradually been evolving postulates the learner's memory structure as an intermediate variable between instruction and learning outcome. Ausubel, Resnick, Scandura, Landa, and Reigeluth have been influential in developing this emerging paradigm (Merrill, Kowallis, & Wilson; 1981). However, thus far cognitively oriented ISD has had little impact on the practice of CAI development. Some of this delay in application is due to the usual resistance to any change in educational practice. However, a more fundamental problem that is emerging is that complex, dynamic models of subject matter, student, and pedagogy are needed to implement cognitively oriented design theories, and CAI is not capable of supporting such sophisticated internal computational representations.

In CAI, the computer does not "understand" the content, student, or pedagogy involved; rather, the device merely presents a preset series of symbols to the learner, which are varied in a predetermined manner depending on the responses the student makes. For simple subject matter, narrow ranges of learner needs, and nonsophisticated mastery outcomes, delineating a set of "frames" presenting material and branches between frames can be a practical approach to meeting instructional objectives. However, for complex subject matters, a wide range of student needs, and sophisticated skills, the number of potential frames and branches undergoes a "combinatorial explosion" into literally billions of possibilities when the concept of possible mental states is introduced (Sleeman & Brown, 1982). This makes a detailed implementation of cognitively based ISD approaches unmanageable in CAI, regardless of the efficiency of the authoring system used to generate frames and branches.

In summary, while no single design methodology dominates the production of CAI materials, historically a strongly behavioristic approach has been reinforced by the types of authoring systems available and the underlying assumptions of CAI itself. This has created an emphasis on "programming" the student into performing a series of low-level skills; the focus is on presenting the sequence of experiences which will lead to optimal acquisition of those skills. Preparing computer-assisted instruction becomes primarily a matter of designing all the possible frames of information which might be required and anticipating all the branches between frames which student needs might dictate.

ICAI: Its Instructional Design and Development Methodology

Like their CAI counterparts, many different types of intelligent tutoring systems (ITS) have been developed over the past ten years. Major applications have included games (WEST, WUMPUS), simulations (STEAMER, SOPHIE), tutors (LISP TUTOR), drill-and-practice (BUGGY), and problem solving (PROUST, PIXIE) (Dede, 1986). Although several different authors (e.g., Clancey, 1986; Sleeman & Brown, 1982) have proposed general methodologies for the development of ITS—expert module, student module, tutoring module, communications module—their approach describes logical structures of existing systems rather than a specific development strategy.

In addition, in attempts to distinguish between traditional CAI and ICAI, careful attention must be paid to distinctions in vocabulary. For example, confusion exists over how to map the terms used to describe ITS into CAI terminology. When authors of ITS discuss "capturing the cognitive processes of students," traditional instructional designers think about "performing a task analysis." ICAI's "adaptive methods of control" and "if . . . then productions" conjure up visions of "branching to remedial areas." These concepts do *not* map onto each other, but CAI developers unfamiliar with AI approaches may conclude that their approach is fundamentally similar. (A detailed comparison of the differences between CAI and ICAI development strategies is presented in Park & Seidel [1987].)

What are the most crucial differences between ITS and traditional CAI? A useful general distinction is that AI-based instructional programs contain dynamic models of the task, the student, and the teaching discourse (Clancey, 1986). Operations are then defined that manipulate these models as the learning situation evolves. For authors of ITS, learning objectives are not expressed as behaviors constructed through elementary stimulus/response associations, but as mental procedures and knowledge structures that are developed and used by the learner.

ITS methodology relies heavily on cognitive psychology for its direction and philosophy. Questions that the ITS designer routinely asks about the student include, "how is information stored, organized, and retrieved?" and "how is new knowledge integrated within existing cognitive structures?" ITS designers then create data structures (knowledge representations) and procedures (inference mechanisms) that simulate human cognitive processes.

The principal difference between traditional CAI's behavioral focus and ICAI's use of unobservable cognitive processes underlying performances, skills, and abilities is that ICAI systems handle not just inputs and outputs, but also understand and purposefully capture (1) the mental dynamics that occur within the student and (2) the progres-

AI-based instructional programs contain dynamic models of the task, the student, and the teaching discourse.

sion of instructional process and tasks. Each module within an ITS is not simply reactive, but models an external process. We next explain why instruction organized in this way has the potential for greater effectiveness and efficiency.

Qualitative Models

"Models" (representations of objects or processes) are a very powerful means for organizing knowledge. A useful contrast can be drawn between "static" models, such as the toystore kits which replicate in plastic and on small scale a plane or a truck, and "dynamic" models, such as a small, operational radio-guided airplane. The static model gives a comprehension of how the parts relate to one another, what role each plays, and how they are interconnected to create the whole machine. The dynamic model goes beyond that descriptive representation to convey procedural knowledge about how the airplane functions and what the capabilities of its interconnected parts are.

Physical models (such as the operational model airplane) and quantitative models (e.g., the inverse square law of gravity) are frequently used to make detailed predictions about dynamic phenomena: when, how fast, how far. However, processes such as reasoning, learning, and communication are not sufficiently well understood to allow the construction of physical or quantitative models. Even though the precision of numeric or analogical approaches is lacking, "qualitative" computer models can facilitate comprehending such ill-defined processes through describing the evolution of their spatial, temporal, and causal relations in a semi-formal manner. For example, a skilled teacher can judge fairly accurately how long a particular student will be able to concentrate on a subtraction task, even though this qualitative model of individual motivation may be much less formal than a quantitative model used to predict exactly when a pot of water will boil.

Designers of AI-based systems attempt to build dynamic models of the content, learner, instructional process, and communications interface. At our present state of knowledge, these models must necessarily be qualitative.

Designers of AI-based systems attempt to build dynamic models of the content, learner, instructional process, and communications interface.

An example of a dynamic model of content and student is Burton's BUGGY, which diagnoses subtraction errors (Burton, 1982). This ICAI program works by generating a series of subtraction problems for the student; if the problems are not answered correctly, the system attempts to diagnose the algorithmic error the student is making. This goal is not unlike that of CAI drill-and-practice programs, but the logical and inferencing capabilities of BUGGY allow a different method of attaining this objective, a method which draws heavily on a simulated model of the cognitive process of subtraction.

To illustrate, the authors of BUGGY represent the algorithm for subtraction with the following rules:

Rule 1: *If* there is no current column and C is the rightmost column,
then set the current column to be C.

Rule 2: *If* C is the current column and Answer (C) is not blank and there is a left-adjacent column, NC, to C
then set the current column to be NC.

Rule 3: *If* C is the current column and Answer (C) is blank and Top (C) < Bottom (C)
then set goal to Borrow (C).

Rule 4: *If* C is the current column and Answer (C) is blank and Top (C) > Bottom (C)
then write [Top (C) – Bottom (C)] in Answer (C).

Given an expanded version of this data structure and an "inference engine," the program can solve subtraction problems in a general way.

In this particular case, the inference engine is a set of procedures that matches part or all of the existing problem state to the *if* portions of the rules. If the *if* portion of a rule is true, then the rule "fires" and executes the *then* procedure of the rule onto the existing problem state. For example, if BUGGY is dealing with the problem 25 – 12 = ?, the system first matches the *if* portion of rule 1 and sets column 1 to C, producing the following state:

```
  C
 25
-12
----
```

The system then matches the preconditions of rule 4 to the problem state and applies that rule to produce this state:

```
  C
 25
-12
----
  3
```

Rule 2 is then invoked, which produces another state, and so on.

The subtraction "model" for this particular problem is represented by r1, r4, r2, r4. By using this approach, the authors are able to capture sequences of mental processes which can be used to solve different types of subtraction problems. Such a system makes it possible to analyze what is happening in the learner's mind given a sequence of attempted solutions to a set of subtraction problems. Then, the algorithmic defect that underlies a particular pattern of student errors can be hypothesized.

For example, a defective rule can be added which states:

Rule 5: *If* C is the current column and Top (C) < Bottom (C) and Answer (C) is blank
then write [Bottom (C) – Top (C)] in Answer (C)

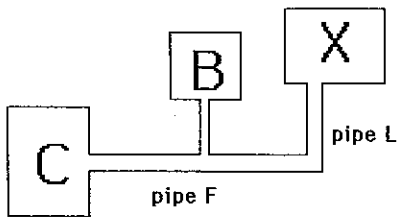
Now, if a student encounters the following problem:

$$\begin{array}{r} 12 \\ -\ 9 \\ \hline \end{array}$$

and enters 7 as the answer, one possible diagnosis available to BUGGY would be the faulty algorithm r1, r5.

Any such sequence of rules represents a process (a rule set) rather than a response to a single question. In this manner, the ITS is able to represent a model of subtraction and multiple models of how a student might systematically err in attempting to solve substraction problems. This is a powerful method of diagnosis based on representing an internal process (dynamic modeling) rather than on decomposing a task into observable behaviors (traditional CAI).

Another example of a dynamic model of content and student can be demonstrated through analyzing a simplified representation of a physical system analogous to that used in SOPHIE (Brown, Burton & DeKleer, 1982), an ITS for learning the diagnosis and repair of electronic systems:



A type of skill that an ITS might teach a student from this display would be how to attempt a repair when Box X goes bad. As with BUGGY, a machine-based expert problem solver could be programmed with a series of procedures which describe how a repair specialist would fix such a situation. The student's actions can then be compared with how an expert might attack the same problem, and the differences between the two approaches noted.

In teaching such problem-solving skills for physical systems, a desirable capability for an ITS would be an option for the learner to query the system about (1) the types of repair interventions that can be made, and (2) which interventions might be most appropriate to consider. A way to represent this problem-solving domain which captures such a capability would be:

SYSTEM: Box X
DISORDER: Leaking
EFFECT: No flow
CAUSES: Electrical short in system
TASK: Test hypothesis (about location of short)
FOCUS: (DISPORDER A$)

If (And (Causes A$ E$)
(Present E$ $System)
then (present A$ $System)

Translation: If a disorder causes an effect and there is evidence that the effect is present in the system being diagnosed, then there is evidence that the disorder is present in the system.

Dynamic representations of this type give the instructional designer a great deal of flexibility. First, the representation can be used to delineate all the different parts of the system to be studied. Second, the representation can be used as a guide (or syllabus) to check whether students understand how each of the parts interact with the system as a whole. Finally, different disorders can be systematically simulated to promote skill building in the student. This approach creates a "glass box" model in which the domain relationships are not static, descriptive axioms, but instead provide a dynamic progression of reasoning about a complex physical process.

Similar dynamic models can be constructed for the pedagogical module and the communications interface of an ITS. For example, a system could be designed to individualize its teaching strategies by making and testing assertions about how a particular student's performance is related to a roster of instructional variables. Over time, such a model would optimize the effectiveness and efficiency of its pedagogical approach through a series of adjustments.

In summary, ICAI systems can be based on sophisticated cognitive models because their architecture includes dynamic, qualitative models of subject matter, student, and pedagogy. These allow the device itself to present material in a variety of ways as broad as the range of possible student mental states, thus avoiding the combinatorial explosion problem which limits the potential usefulness of CAI.

Methodology

A survey of existing ITS systems suggests the following emerging development methodology:

Develop a model of the instructional task (the expertise module). AI programming methodology provides a new means for modeling both processes in physical systems and problem-solving procedures which embed expertise in ICAI devices. In contrast to traditional CAI programs, which have static, descriptive models, these dynamic models are executable and used to solve the same problems presented to the student.

These procedural models are primarily qualitative rather than quantitative, describing agents and actions by their causal, spatial, and temporal interaction. When subject matter is present in this "glass box" form (as opposed to the "black box" of traditional CAI), the dynamic model can both serve as a source of diagnostic expertise and provide the basis for explanatory discussions (Clancey, 1986).

Develop the student model. Many ITS designers use the glass box model in the expertise module to create a dynamic representation of what the student knows. For example, in the physical system described earlier, when the student can correctly identify what is wrong with Box X, this provides diagnostic information about what general cause/effect relationships the student now understands. However, modeling what the student does not know or misunderstands is also important. Therefore, the subtraction system has a repertoire of "buggy" rules on which it can draw to comprehend a student's faulty problem-solving strategy.

Combining the knowledge contained within an expertise module with a library of "bugs" typical of the learner population provides a very powerful method of error diagnosis and remediation. This approach is possible because the student model, like the expertise module, is a dynamic, executable representation—a glass box of the learner's thought processes.

Design the pedagogical module. ITS designers use this diagnostic information from the student model to activate

an embedded planner of instructional strategy. If this planner is represented as a dynamic model, then manipulation, change, and optimization of the pedagogical approach are much easier than with a conventional CAI program. Using the physical system example as an illustration, a teaching rule could be designed which states: "*If* the student does not understand the cause of a particular error, *then* intervene by demonstrating the operation of the part of the system which has failed." Such a model also allows the designer to express higher order instructional strategies such as: "*If* the student consistently gets information concerning causes and effects confused, *then* display a simpler system."

Principles from instructional design theories can be used as heuristics to guide the development of each component in this ITS architecture, particularly the pedagogical module.

Emerging Issues in Instructional Design

Instructional design theory is gradually shifting from a behavioral science orientation to an emphasis on cognitive science, that is, from promoting students' overt performance in manipulating instructional materials to enhancing their cognitive processing. In turn, theoretical approaches to the development of instructional systems are shifting toward directing students' mental processing and interaction rather than manipulating the instructional materials to be presented. However, most current CAI applications are still based on behavioristic ISD approaches, and the limits of CAI's fundamental architecture in implementing cognitively based theories are largely unrecognized by instructional design practitioners.

In part, the failure of "conventional wisdom" in ISD to differentiate between CAI and ICAI as fundamentally different can be traced to a deeper problem in instructional design. Historically, the tendency in this field has been to emphasize the competitive aspects of different models rather than to view each approach as a complementary strand in a general theory (Reigeluth, 1983). Some instructional designers have pushed for their model to be *the* theory, even though its assumptions and methods may be optimal only for a specialized type of content, teacher, or pupil. As a result, rather than seeing CAI and ICAI as alternative approaches, each needing its own design strategy, models tailored to human or CAI instructional capabilities have inappropriately been generalized to ITS applications.

A theory of instruction must have three major components: methods, conditions, and outcomes. Instructional *conditions* are defined as variables that (1) interact with methods to influence their effects on outcomes, and (2) cannot be manipulated in a given situation (Reigeluth & Merrill, 1979). The potential strengths and limits of the "teacher" (whether human, CAI device, or ITS) are important conditions often underemphasized in instructional design models. In particular, CAI and ICAI systems have quite different instructional attributes which dictate the use of divergent ISD models having methods and outcomes tailored to the characteristics of each (Halff, 1986).

For example, existing ISD approaches are structured to be used either by intelligent human designers or by mindless machines. Pedagogical heuristics are stated either in very general form, relying on people's common sense and peripheral real-world knowledge to determine appropriate applications, or in branching frame formats, since CAI devices must work on this specific level. The different instructional attributes of ITS require an intermediate representation of heuristics. Also, ISD has not historically emphasized intelligent tutoring, i.e., using meta-rules for global evaluation of the instructional context and dynamic modification of the pedagogical plan (Woolf & McDonald, 1985), yet this is a strength of ICAI applications.

Halff (1986) lists emerging research issues important in the evolution of instructional design for ITS. These include:

- research on the strengths and weaknesses of alternative tutoring approaches
- experiments with "Wizard of Oz" systems in which a human tutor simulates some of the functions of an automated tutor during the design process
- theories of learning that embody internal symbolic representations
- models of communication that involve joint interpretation by instructor and student
- research on the "modularity" hypothesis of making instructional and domain knowledge independent
- experimentation with intermediate (non-expert) representations of subject matter as a bridge between novice and skilled performance

Dede (1986) delineates emerging questions for research, which include:

- What level of procedural and declarative justification must be embodied in an expertise module to produce explanatory capabilities?
- How should the learning of processes be sequenced to maximize links to previous procedures and interconnection into an overall framework of semantic rationalization?

Theoretical approaches to the development of instructional systems are shifting toward directing students' mental processing and interaction....

- How can problem solving be structured to minimize the need for pupils to attempt repairs to a faulty set of heuristics?
- To what extent can intervention criteria (such as relevancy or memorability) be assessed independent of subject matter or student attributes?
- Within what limits can typologies of explanation, theories of hints, approaches to example selection, or rhetorical strategies developed for a particular subject be generalized to other domains?
- What proportion of total instructional effectiveness can be attained via self-improvement through machine learning?

The emergence of theoretical initiatives to develop new pedagogical models tailored to the attributes of ICAI systems will be an important step in the evolution of an overall theory of instructional design. Research targeted to issues such as those above will be most useful.

References

Bork, A. (1981). *Learning with computers.* New York: Digital Press.

Bork, A. (1986). *Nontrivial, nonintelligent computer based learning.* Irvine, CA: Educational Technology Center, University of California—Irvine.

Briggs, L. J. (Ed.). (1979). *Instructional design.* Englewood Cliffs, NJ: Educational Technology Publications.

Brown, J. S., Burton, R. R., & de Kleer, J. (1982). Pedagogical, natural language, and knowledge engineering techniques in SOPHIE I, II, and III. In D. Sleeman & J. S. Brown (Eds.), *Intelligent tutoring systems.* Hillsdale, NJ: Lawrence Erlbaum Associates.

Burton, R. R. (1982). Diagnosing bugs in a simple procedural skill. In D. Sleeman & J. S. Brown (Eds.), *Intelligent tutoring systems.* Hillsdale, NJ: Lawrence Erlbaum Associates.

Carbonell, J. R. (1970). *Mixed-initiative man-computer instructional dialogues.* Ph.D. thesis, MIT, Dept. of Electrical Engineering, Cambridge, MA.

Clancey, W. J. (1986). Qualitative student models. *Annual Review of Computer Science.*

Dede, C. J. (1986). A review and synthesis of recent work in intelligent computer-assisted instruction. *International Journal of Man-Machine Studies, 24,* 329-353.

Duchastel, P. (1987). *Models of learning in ICAI.* Quebec, Canada: Universite Laval.

Gagne, R., & Briggs, L. (1977). *Principles of instructional design.* New York: Holt, Reinhardt, and Winston.

Halff, H. M. (1986). *Curriculum and instruction in automated tutors.* Brooks AFB, San Antonio, Texas: AFHRL Workshop on Intelligent Tutoring Systems.

Keller, A. (1987). *When machines think: Designing computer courseware.* New York: Harper & Row.

Lewis, M. W., Milson, R., & Anderson, J. R. (1986). Designing an intelligent authoring system for high school mathematics—ICAI: The Teacher's Apprentice Project. In G. P. Kearsley (Ed.), *Artificial Intelligence and Instruction: Applications and Methods.* New York: Addison-Wesley.

Merrill, M. D. (1983). Components display theory. In C. M. Reigeluth (Ed.), *Instructional design theories and models.* Hillsdale, NJ: Lawrence Erlbaum Associates.

Merrill, M. D. (1985). Where is the authoring in authoring systems? *Journal of Computer Based Instruction, 12*(4), 90-96.

Merrill, M. D., Kowallis, T., & Wilson, B. G. (1981). Instructional design in transition. In F. H. Farley & N. J. Gordon (Eds.), *Psychology and education: The state of the union.* Berkeley, CA: McCutchan Publishers.

Park, O., & Seidel, R. J. (1987). Conventional CBI versus intelligent CAI. *Educational Technology, 27*(5), 15-21.

Reigeluth, C. M. (1983). Concluding remarks. In C. M. Reigeluth (Ed.), *Instructional design theories and models.* Hillsdale, NJ: Lawrence Erlbaum Associates.

Reigeluth, C. M., & Merrill, M. D. (1979). Classes of instructional variables. *Educational Technology, 19*(3), 5-24.

Russell, D. M., Moran, T. P., & Jordan, D. S. (1986). *The instructional design environment* (Technical Report). Palo Alto, CA: Xerox Palo Alto Research Center.

Scandura, J. M., Stone, D. C., & Scandura, A. B. (1986). An intelligent rule tutor CBI system for diagnostic testing and instruction. *Journal of Structural Learning, 9,* 15-61.

Skinner, B. F. (1953). *Science and human behavior.* New York: Macmillan.

Sleeman, D., & Brown, J. S. (Eds.). (1982). Introduction. *Intelligent tutoring systems.* Hillsdale, NJ: Lawrence Erlbaum Associates.

Snelbecker, G. E. (1983). Is instructional theory alive and well? In C. M. Reigeluth (Ed.), *Instructional design theories and models.* Hillsdale, NJ: Lawrence Erlbaum Associates.

Steinberg, E. R. (1984). *Teaching computers to teach.* Hillsdale, NJ: Lawrence Erlbaum Associates.

Tennyson, R. D., & Christensen, D. L. (1986). *Memory theory and the design of intelligent learning systems.* Minneapolis, MN: School of Education, University of Minnesota.

Woolf, B., & McDonald, D. D. (1985). Building a computer tutor: Design issues. *AEDS Monitor, 23*(9-10), 10-18.