Contents

ARTICLES

Research, Instructional Design, and New Technology
Kathie Fletcher Alred and Craig Lofquist  2

A Four-Stage Model for Planning Computer-Based Instruction
Gary R. Morrison and Steven M. Ross  6

The Evolution of Instructional Design Principles for Intelligent Computer-Assisted Instruction
Christopher Dede and Kathleen Swigger  15

Needs Assessment and Analysis: Tools for Change
Stephen R. Rodriguez  23

The Intellectual Content of Instructional Design
Wayne A. Nelson, Susan Magliaro, and Thomas M. Sherman  29

DEPARTMENTS

Book Reviews, edited by Allison Rossett
Training Needs Assessment by Allison Rossett
Reviewed by Judith Fidler  36

ERIC Reports on ID, edited by Barbara Minor  38

CORRECTION

The previous issue of JID, Vol. 10, No. 4, was dated 1988. It should have been dated 1987, as was indicated at the bottom of each page inside the journal. The publisher apologizes for the error.
Research, Instructional Design, and New Technology

Kathie Fletcher Allred
Hughes Associates, Inc.
Craig Locatis
National Library of Medicine
National Institutes of Health

Research provides valuable guidance for formulating instruction, but using results can be difficult. Research findings often are obtained under controlled conditions, while instruction usually occurs in less controlled settings. Moreover, research isolates variables, while instruction integrates them. In addition, research often produces discrepant outcomes, apparent and real, that must be resolved in application. For example, studies show that learner control facilitates intrinsic motivation, but can be suboptimal for learning. Yet intrinsically motivated learners tend to achieve more. Finally, there is seldom an exact match between real-world instructional problems and those addressed in research. The amount and kind of practice and feedback used in a given situation involves judgment, even though research has documented their benefits. New technology exacerbates these problems because design formats, issues, and questions emerge that researchers have not yet addressed.

Three instructional designs involving new interactive media are examined in this article. Hypotheses about their learning effects are advanced, based on findings in three areas of research. The designs—adaptations of other information technology applications—were chosen because they tend to utilize interactive media capability fully. The research areas were selected because they address some of the benefits claimed for each design. The analysis was undertaken because there is little research on the designs, and because existing instructional research in other contexts may have something to say about them.

The focus here is on the instructional designs that new technologies allow, rather than on the media themselves. Research comparing media often is not generalizable and usually confounds variables (Clark, 1983). Studies of computer-based versus classroom instruction, for example, typically find no performance differences, but time savings for those who were computer taught. However, the time reduction is probably due to individualized instruction, not to computers. Moreover, computer-based lessons take longer to develop and often incorporate instructional strategies promoting efficient learning.

Designs

Early interactive programs were pedestrian. Many videodiscs presented lectures, and microcomputer courseware mimicked much of the electronic "page turning" deplored on larger machines. Teaching strategies were used which were known to be ineffective in other, related media. When microcomputers and videodiscs were used together, even the best instructional designs often failed to integrate their capabilities. One medium would dominate because course developers seldom were familiar with both. Programs appeared with long, linear video episodes and little interactivity, or with a lot of interaction and a few poor-quality video images stuck in. Recently more imaginative designs have emerged showing how linking microcomputers and videodiscs makes possible learning environments that are more than the sum of their parts. These designs can be applied to other information technologies such as interactive and read-only compact disks. The designs can be categorized as scenario-based, hypermedia, and parallel systems.

Scenario-based Designs

These designs expose learners to scenarios that are the basis for more didactic instruction. The scenarios are realistic or fantasy "worlds" of video and graphic images. Learners usually are given a goal and must perform tasks to achieve it. A range of tasks and decision options is provided; performance while interacting is monitored in the background. Remedial instruction is given either at the end of the scenario, if the goal is not attained, or during interaction, if too many errors are made. The interface makes interaction easy, often through the use of touch screens, voice recognition units, or other input devices.

There is a series of videodisc emergency medicine simulations that exemplify scenario-based instruction. These programs are used on special workstations having microcomputers, videodisc players, and touch-screen monitors. Physicians view different video episodes of a patient whose condition changes over time. They gather information and prescribe therapies by touching selections on the screen. When the simulations end, physicians can repeat them, see an explanation, compare their decisions to those of experts, view highly structured lessons explaining the condition portrayed, or access references (Allen, 1986). Another example is the army's STARS
Each display in the knowledge base functions as a "menu" to other displays.

Hypermedia Designs

These designs provide learners with knowledge bases and related retrieval tools. Original information sources are on-line, not just references. These sources may be textual, graphic, pictorial, or some combination. Each display in the knowledge base functions as a "menu" to other displays and contains cues to additional information available. Learners use these menus to move and manipulate information displays, mark information for later retrieval, reference personal notes, or search related topics. Movement can be both within and among different knowledge bases. Hypermedia programs have been proposed where documents and images could be copied and modified to encourage joint authorship. Trails kept on these transfers would show the evolution of ideas and enable royalty payments (Nelson, 1983).

An example of a hypermedia design is an electronic textbook on pathology (Thurst & Mabry, 1980). Highlighted terms indicate availability of additional text or images. Pop-up menus are evoked by positioning the cursor on terms. Learners can choose to view images on videodisc, see definitions, or search for the term or related topics in the text. The idea is eventually to have an electronic bookshelf so that learners studying the pathology text might instantly jump to anatomy and other textbooks for related information. A second example is an experimental program teaching pushcart assembly (Stone & Hutson, 1984). The initial menu is simply a graphic of the entire cart. Positioning the cursor on any part of the cart causes other displays of assembly diagrams and/or written directions to appear. A third example is a visual data base in histology (Bridgman, Telford, & Allen, 1985). Bar codes are embedded in a hardcopy textbook so its pages can become menus to other information. A bar code reading wand on a small computer is used to call up related videodisc images that augment the text. A visual data base with a different interface overlays icons on video images. Learners use a mouse to select icons that let them see pictures at greater magnification or from different angles (Wertheim, 1986).

Parallel System Designs

These designs involve constructing complementary teaching and testing programs around a common knowledge base. The knowledge base can be accessed in multiple modes, such as information, instruction, or test. For example, in information mode, text and visuals are browsed or retrieved. In instruction mode, additional pre-designed information and teaching strategies are provided. In the test mode, questions about knowledge base content are presented. The knowledge base, instructional strategies, and test questions reside "alongside" each other. They exist as separate computer files, but are cross indexed so users can shift between modes any time. One user might start exploring a topic in information mode, decide to be tested on it, and, depending on results, opt for instruction. Another might choose to be tested on the entire knowledge base, and then review only certain topics by accessing information or being taught.

To the authors' knowledge, systems that have been designed intentionally for use in either "tell me," "teach me," or "test me" modes do not exist. One existing system, however, has some of these features. A/Rheum is an expert system that diagnoses rheumatic diseases. Physicians and medical students can submit data for a new case or choose a case from the system's library. The system renders multiple diagnoses or indicates that it lacks sufficient information. Findings supporting each diagnosis are displayed; additional information needed to confirm the primary diagnosis or an alternative is indicated. Users can ask the system to display the rules used to reach a decision or show and tell them more about diseases and clinical findings. Additional text or videodisc images are displayed in response to these requests (Kingsland, Lindbergh, & Sharp, 1986). Moreover, a complementary system, A/Learn, is being developed that uses the expert system's videodisc images and decision rules to teach sign and symptom recognition and clinical reasoning.

Research

At least three advantages for using these designs have been advanced. One is that the designs are more motivating—they engage learners and make content more interesting. Another is that the designs may be more suitable for those who have difficulty learning by more traditional methods. A third is that the designs have the ability to give individuals more control of the learning process. Research on intrinsic motivation, aptitude for learning, and learner control of instruction can clarify the validity of these claims.
Intrinsic Motivation

Lepper and Malone (in press) identified three motivation inducers in their research review of intrinsic motivation. Each reflects a different research tradition and view of human nature. The inducers involve providing activities that are neither too easy nor too difficult, that are novel (neither too foreign nor too familiar), and that give some control and self-determination. The first, which considers humans to be problem solvers, is based on research on challenge and feelings of accomplishment. The second, which considers humans to be information processors, is based on research about curiosity and discrepancy coping. The third, which considers humans to be voluntary actors, is based on research about locus of control and feelings of efficacy.

A fourth inducer was not described specifically, but might be inferred from Lepper and Malone’s extensive analysis of effective computer game features and the examples they present. This inducer would be to evoke a playful “set,” since the games encourage interaction in a nonthreatening context and provide performance feedback. These designs might assume humans to be activity seekers who try to identify personally with their environments.

Aptitude for Learning

Snow and Lohman (1984) identify twelve major findings in their research review of aptitude for learning, three of which are pertinent to the three instructional designs. First, high-ability learners can perform assembly and control functions for organizing activities and guiding learning that low-ability learners cannot. Second, high-ability learners tend to perform better when the instruction provided is incomplete and not prescribed precisely. Since their assembly and control functions are not only more refined but also more idiosyncratic, they adopt highly individualized learning strategies based on their own unique knowledge, background, and experience. Third, providing training and support in self-learning strategies can aid low-ability learners but may not help those of high ability, since high-ability learners’ finely tuned learning techniques might be replaced with ones that are less personally optimal. Snow and Lohman warn that teaching strategies that are mathemagician (giving birth to learning) for some can be mathemagician (giving death to learning) for others.

Ideally, instructional treatments should vary in completeness and teaching support, and low-ability learners should be taught self-learning skills. Ironically, both high- and low-ability learners tend not to enjoy the teaching strategies that are most effective for them (Clark, 1982).

Learner Control of Instruction

Tennyson and his colleagues (Tennyson, Christensen, & Park, 1984) identify conditions where learner control of instruction may be effective. They have compared performance in learner-controlled and adaptive teaching programs where instructional events change dynamically based on a learner’s performance history. For example, the number of practice exercises might vary for each learner when an algorithm using Bayesian statistics is employed to monitor performance and determine whether additional instruction is required.

Adaptive programs are superior to those giving learners total control. However, adding advice improves performance in both programs. Moreover, providing advice in a learner-controlled program can be more efficient than using an adaptive program without advice. Subsequent learner control studies indicate that when learners are given advice, they tend to take it (Hannaftin & Colamalu, 1986). If this advice is to adopt the same instructional strategy used in an adaptive or teacher-controlled program, research treatment differences are eroded. However, the effects of advice on experiment integrity are less disturbing than Snow and Lohman’s suggestion that advice may have deleterious consequences for high-ability learners.

Implications

Each area of research provides varying degrees of support for the instructional designs, both individually and collectively. The research suggests when the designs might be used most appropriately with different learners.

- Research on intrinsic motivation would tend to support each approach because they all provide environments that are novel, challenging, self-directed, and sometimes playful. However, if the scenario, hypermedia, and parallel system environments are so complicated that learners lack the requisite knowledge and skill to interact effectively, they could become demotivating. Scenario-based instruction offering remediation when performance is poor might avoid this problem. Unless learners are carefully matched to the instruction on the basis of their existing knowledge and skill, demotivation remains a potential problem for each design.

- Aptitude for learning research also supports each design, but only for sophisticated learners. Most of the designs are open, incomplete, and allow use of idiosyncratic learning tactics.
Hypermedia environments provide little or no teaching support other than the content presented. These conclusions only apply to the designs as they are presently conceived and implemented. Modifications might rectify their deficiencies. For example, scenario-based instruction might be devised that provides not only remediation, but also advice, and that adjusts the complexity of a situation a learner encounters. Parallel systems might have both sophisticated and simplified explanations of content in their knowledge bases, as well as multiple teaching strategies. They could advise learners about which explanations and teaching strategies to use. Similar features might be built into hypermedia programs. These adjustments and enhancements might ensure successful interaction and learner motivation, and make the designs more appropriate when self-learning skills are lacking. Perhaps the amount of additional support provided could be adjusted according to learner sophistication. On the other hand, such modifications may be unnecessary or undesirable for high-ability learners.

Conclusion

The instructional designs described have the following common characteristics: they are information rich; they provide unprecedented amounts of learner control over what content will be learned and the sequence and pace of instruction; and the control is highly virtual, with the technology responding almost immediately to learner requests. If these designs are indicative of others that might be realized, then the new information technologies make these designs become increasingly incorporated into the general information technologies people encounter in everyday life, they underscore the need to teach people better self-learning skills.

References

A Four-Stage Model for Planning Computer-Based Instruction

Gary R. Morrison
Department of Curriculum and Instruction
and
Steven M. Ross
Foundations of Education
Memphis State University
Memphis, TN 38152

Abstract. Computer-based instruction (CBI) lessons require thoughtful design and careful planning during their development. Existing CBI development models are not adaptable to lessons that vary in complexity or to differing roles and skill levels of a project staff. A flexible planning process in which the CBI design is implemented on paper as an intermediate step between the lesson design and the program production is described in this paper. The process consists of four major components: an initial flowchart, storyboards, a detailed flowchart, and an evaluation. Each step is described in detail using examples from recent applications. Emphasis is placed on the adaptability of the system for accommodating CBI projects that vary in scope and orientation.

steps from the identification of the instructional problem through the design of the instructional strategies. At a minimum, the design would prescribe the sequences and form of interaction, the type of feedback, and the use of graphics and sound. Lesson authoring begins once the design is completed and involves writing the computer code to generate the lesson. Different approaches to the instructional design process are well documented in the literature (e.g., Jonassen, 1988). There is less documentation, however, on the process of translating into a CBI lesson the content defined during the instructional design phase.

There is a need for a well-defined but adaptable planning model for translating instructional design plans into CBI lessons. Such a planning model should meet four criteria. First, it should be adaptable—the model should operate as effectively for designing a simple drill-and-practice program as for a complex simulation. Second, it should provide an efficient means of "prompting" the designer for critical information about branching, data storage, cues, graphics, and sound (Richards & Salisbury, 1987), as well as for the components of the instructional design model. Third, it should provide a clear and accessible view of both the structure of the lesson and the amount of learner interaction. Fourth, it should provide a means for identifying various modules and subroutines to simplify the lesson authoring. A model that has been developed in accord with these four criteria and that has been used in several recent projects is described in this article.

Current Planning Approaches

The most common approach to CBI design involves using grid sheets (a 1:1 representation of the computer screen display) to specify content and display information (Richards & Salisbury, 1987). Each display of a lesson is represented as a separate storyboard in much the same way as scenes or slides are represented on video and slide storyboards. This approach, when applied to CBI displays, concentrates solely on the screen displays while tending to ignore the structure of a les-

There is a need for a well-defined but adaptable planning model for translating instruction design plans into CBI lessons.
son (Bork, 1985; Richards & Salisbury, 1987). One of the major criticisms of Richards and Salisbury (1987) specifically note is the lack of cues on the screen design grid to prompt designers to maximize specific computer attributes such as graphics, interactivity, and animation.

Several recent planning models employ grid-based systems. In Allen and Erickson's (1986) interactive videodisc design model, the grid allows the designer to specify the exact placement of the content, prompts, graphics, and other visual or textual information. In Allessi and Trollip's (1985) model, the design of an instructional sequence is followed by the design of screens using a grid-based system. Dean and Whitlock (1983) use a similar grid design for screens, but they also develop a mainline chart that presents the content of the frames in the most direct path through the lesson, and a flowchart that indicates both the direct path and specific branches available to the learner. The Dean and Whitlock model provides for flexibility, but is redundant by repeating the content of the primary frames in both the mainline chart and the storyboards.

As an alternative to the grid sheet, Bork (1985) has proposed the development of a “script” that incorporates screen designs into individual flowchart blocks. The scripting process, however, can be cumbersome for the programmer to translate into screens since it is done in free form, i.e., it does not match the size constraints imposed by the CRT screen.

Another alternative to the grid procedure is Richards' and Salisbury's (1987) Screen Design Syntax (SDS). SDS involves using a code to identify such items as text, graphics, branches, audio, and correct and incorrect answers (see Figure 1). This syntactic code is similar to the pseudo-code used in programming. SDS's main advantage is the use of a word processor for formatting text, making similar or duplicate frames, and facilitating revisions.

SDS has four limitations. First, graphics cannot be incorporated directly into the screen design. Second, there is no coordinate system, as is available with grid sheets, to indicate the exact location of the text or graphic. Third, designers must master the special syntax used for designing the screens. Fourth, SDS fails to
adequately overcome one of the major limitations of grid designs—the absence of prompting for design considerations (e.g., graphics, animation, highlighting) as the screens are designed.

A Working Model for CBI Planning

Our planning process for translating an instructional design into CBI lessons includes four steps (see Figure 2). First is the development of a general flowchart identifying the major components of the lesson. Second is the development of the storyboards or screen designs with information on the variables and direction of program flow. Third is the development of a detailed flowchart from the storyboards. Fourth is the evaluation of the design plan and the resulting program. A description of each step follows.

Initial Flowchart

For the CBI designer, an initial flowchart is similar to the table of contents of a book. It specifies the major program components and their general organization. For example, the initial flowchart for a simple lesson might identify the title screens, a lesson menu, lesson segment 1, lesson segment 2, data storage, and instructional management components. In designing the initial flowchart, simplicity rather than detail is the key. Three symbols—a rectangle, a diamond, and a circle—are usually adequate for depicting program components. A rectangle is used to represent major sections of the lesson, such as the introduction, instructions, rules, examples, tests, etc. More elaborate print formatting or data storage instructions, if required, are provided on a separate flowchart constructed by the programmer. A diamond is used to indicate answer judging and decision points for program branching. A circle indicates continuation from one page or section to another. Brief descriptive labels are used to identify each component on the initial flowchart. To illustrate, Figure 3 shows a simple flowchart for a CBI unit consisting of three lessons and a posttest.

Initial flowcharts serve three purposes. First, they display the major segments or parts of a lesson. Second, they allow a designer to evaluate the flow from segment to segment and determine what transitions are needed between lessons. For example, at the end of a segment, should the program return to a menu or simply ask if the learner is ready to proceed to the next segment? (In the present example, the program returns to the menu following the first and second lessons, and branches to a posttest option following the third.) The third purpose is the identification of the major lesson components for use in scheduling, budgeting, and division of work if more than one programmer will be working on the program.

Once the major components and program flow are determined, screen design can begin, making use of the storyboards.

Storyboard

A storyboard is used with different media such as slides, film, and video to plan the individual scenes of a production. Similarly, a storyboard can be helpful for planning individual frames of a CBI lesson. The storyboard usually consists of a grid representing the CRT screen, frame identifiers, and space for notes. This 1:1 correspondence between the screen and storyboards enhances the efficiency of the storyboard for developing screen designs, translating the designs into code, and evaluating the CRT screens for accuracy. By incorporating a screen grid and related lesson specifications, the present model prompts designers for five types of information on each frame (see Figure 4). Depending on the specific lesson design, all or only some of these prompts may actually be used. A description of each prompt follows.
Frame Numbers/Identifiers. The first component on the storyboard identifies the number and type of the current frame and of previous contiguous frames. Frame numbers are used to track the flow of the program when making a detailed flowchart. An example of identifiers used in this model are illustrated in Figure 5. Similar identifiers can also be included in REM statements or as labels in the program code to help identify individual frames during debugging and revisions, or they may be printed to the screen to aid program debugging and to check the program's logic. The numbering syntax also serves to identify the following six general frame types.

1. Information frames are indicated simply by a frame number (1, 2, 3, and so on).

2. Question frames are indicated by the frame number followed by the letter Q (17Q, 38Q, etc).

3. Remediation frames for individual questions are indicated by the associated question frame number and the

Figure 4. Screen design grid.

Figure 5. Example of frame numbering scheme.
These six general frame types were selected as representative of most frames used in computer-based instructional programs. Individual designers, however, might want to revise or create new frame types to match the components of their instructional strategies. For example, one could devise a notation for preinstructional strategies, example and nonexample concept frames, and coordinate concepts. Selecting frame identifiers that more closely reflect the components of the instructional strategy could also aid in the evaluation of the sequence. For example, a designer using the component display theory (Merrill, 1983) might adopt a syntax oriented to that model to reflect the prescribed primary and secondary presentation types.

**Screen Grid.** The second component of the storyboard is the screen grid which consists of 40 or 80 columns and 24 rows. This grid is used to design individual frames and templates for repetitive frames. For a simple drill-and-practice lesson, such as one involving identification of map outlines of the 50 states, one or two storyboard templates could specify the location of the map outline, the question stem, learner input area, and prompt area. All 50 drill frames could then be coded by the programmer from the template. Individual frames containing unique formats or content are designed on individual screen grids with text and graphics entered exactly as they are to appear on the screen. The resultant 1:1 correspondence between the grid and the CRT screen allows the programmer to easily understand the intentions of the designer.

Minor changes in subsequent frames or windows are indicated by drawing a box on the new frame and noting the old information (and reference frame) that stays on the screen. Information for the new frame is entered on the grid in the appropriate parts.

**Branching Instructions.** The third storyboard component is the branching notation that directs the lesson flow according to a fixed sequence or a variable sequence based upon learner responses. Branching locations are identified for each possible response to questions, menus, and other frames.

---

**Figure 6.** Example CBI frame with animation instructions.
requiring input, including simple keypresses (e.g., space bar, ESCAPE) used for forward and backward paging. Example notation would be of the type, "On A goto frame 25R. "On B goto M1," "On ESCAPE goto END."

Designating Variables. The fourth component is information on what learner response data are to be stored on a disk or to be used as variables within the program. Designers and programmers must identify these variables and the frames at which they are to be established. The model addresses this need by providing two types of prompts. The first prompt, labeled "STORE RESPONSE AS," is used to store either the student's actual answer or information about the answer (e.g., Right or Wrong). The second prompt, "SET FLAGS," is used to increment counter variables for tracking such data as the number of correct and incorrect responses, the number of times a frame is viewed, or the number of attempts required to answer correctly. Conditions under which credit is to be awarded (e.g., first try only) or when branching to a help frame is to occur (e.g., after the third wrong attempt) are also specified as flags.

Special Instructions. The fifth component of the storyboard is "special instructions" regarding uses of features such as color, animation, sound, or inverse or flashing text. This component is also used to specify information such as timing variables, notes on the placement of windows, replacement of text, and other special screen or coding features. Figure 6, for example, shows the special instructions specified for an animated sequence in a program that was developed to simulate a Piagetian interview (Ross, Barnette, & Morrison, 1987).

Detailed Flowchart

The next step in the process involves the design of a detailed flowchart from the information specified in the completed storyboards. The rectangle, diamond, and circle used in the initial flowchart are also adequate here, since the primary purpose is to identify program flow as opposed to the specific programming statements (e.g., READ, PRINT, etc.). In the detailed flowchart, a rectangle roughly corresponds to a single frame. The finished flowchart is used to evaluate the program design, develop code, and evaluate the completed software.

Development. Figure 7 presents parts of a detailed flowchart used to represent a drill-simulation program on "empathetic listening" (Ross, Barnette, & Morrison, 1987). Development of the flowchart starts by plotting the first frame as a rectangle with the frame number inside (notes or titles for reference purposes are often included with the frame numbers). Next, it is determined whether the lesson's logic allows branching to several frames or only to one frame. Where branching options exist, as in question or menu frames, a diamond symbol is used as the decision point and rectangles as the specific options. Again, the associated question frame number is written in each diamond. Circles may also be used to identify the beginning and end of each page of the flowchart and the sequences of pages. Any time a subroutine is used, a circle with the subroutine number is connected to the rectangle or diamond. The next section describes how the storyboards and detailed flowchart are used in the formative evaluation process.

---

Figure 7. Example of a detailed flowchart.
Evaluations provide the designer with a means of evaluating the lesson design before time and money are invested in the coding process.

Evaluation of Program Design

The completed storyboards and detailed flowchart are used to conduct four formative evaluations of the instructional sequence and the resulting program. Three of the evaluations are completed before the programming begins. The fourth is an evaluation of the CBI lesson.

Design Evaluation. Both the designers and subject-matter experts participate in the design evaluations. The first evaluation analyzes the sufficiency of branching instructions and verifies their conformity with the storyboards. Corrections in the sequence or the addition of new frames are made during this stage. For example, it may be necessary to insert a review option at the completion of an instructional segment, add an option to allow experienced users to bypass the instruction frames, or decide to introduce a prompt as a window. This initial analysis thus serves to identify weaknesses in the program design that can be corrected before the coding process begins.

The second evaluation identifies situations in which unanticipated responses could occur. If, for example, the user is prompted to enter the date in the form “MM/DD/YY,” what will happen if the slashes are omitted, one digit is entered instead of two, or an unrealistic value (13 for month) is entered? This analysis may suggest the need for special help frames that give additional procedural instructions, or for special error trapping routines.

The purpose of the third evaluation is to determine the amount of interactivity in the lesson. Is the activity primarily “page turning,” or does it involve answering questions, solving problems, or making content-oriented decisions? A primary focus of this evaluation is interaction that stimulates conscious cognitive processing, as described by Merrill (1983). Jonassen (1985) also provides a taxonomy that can serve as a basis for an instrument to assess the levels and types of interaction in the lesson. Infrequent branching suggests ineffective use of the computer’s interactive and adaptive capabilities.

These three evaluations provide the designer with a means of evaluating the lesson design before time and money are invested in the coding process. Additional frames can easily be added at this stage without incurring additional costs for revising computer code.

Product Evaluation. The fourth evaluation uses the storyboards and detailed flowchart to assess the accuracy of the coding and screen designs. During the product evaluation, the designer determines whether the branching instructions follow the specifications indicated on the storyboards and flowchart, and whether the CRT screen designs are effective with regard to such qualities as (a) liberal use of “white” space around graphics, blocks of text, and between lines, (b) clear visual separation of response prompts and lines of other text, (c) clear labels and prompts, (d) appropriate use of inverse or flashing words, (e) proper grammar and spelling, (f) split words, and (g) overwriting of graphics by the text. Color, inverse words or lines, and general layout may need to be adjusted for aesthetics or impact after the evaluation. Specific criteria for evaluating screen design can be developed from the works of Fleming and Levie (1978), Heines (1984), Hooper and Hanafin (1986), and Schneiderman (1987).

Storyboards and a detailed flowchart provide an accurate and efficient means for evaluating and revising the materials during the design phase and during the coding phase.

Applications and Utilization

This section describes different applications of the CBI planning model. Applications include using the storyboard grid designs to develop screen templates, to improve the human interface, and to evaluate the program design. Some strategies to facilitate programming are also described.
Screen Templates

The screen grid (Figure 4) provides a means for designing standard templates for lesson frames. Templates help ensure that the materials produced by different designers and programmers are consistent throughout the project. One of the first steps in designing a standard screen template is to determine the location of standard components—status line, prompt area, and learner response (Heines, 1984). These three components should be displayed in the same relative location and manner in each frame and across related lessons.

Figure 8 presents a sample screen based on a standard template. The status line, shown in the top row, typically identifies the location in the lesson (frame 12 of 23), the title of the module or segment (Piaget I), and type of frame (question, information, etc.). Icons such as a thermometer or clock can also be used to indicate progress or location in a lesson.

The second standard component is the prompt area (bottom of Figure 8) which communicates what action the learner should take, and the acceptable responses to enter. For example, the prompt line might direct the learner to press Q to quit, R to review the content, or A, B, or C to choose an answer.

The third component is the learner response area (middle of Figure 8). Heines (1984) recommends that a standard area be designated for user input, such as space below the multiple-choice alternatives. The designer might also choose an icon or phrase to indicate that area. For example, two dashes and an arrow may be used to prompt the learner for input to questions and menus. For fill-in-the-blank type questions, embedding a familiar prompt/cursor within the sentence itself, rather than at the bottom of the screen, avoids the situation in which the learner must glance back and forth from the sentence to the input area. Most importantly, the status line, prompt line, and input area should always appear in the same screen locations, and use the same symbols across frames to provide consistency both within and between modules.

Interface considerations

An additional area to consider for standardization is the user interface. The designer may want to specify the parameters for menus, multiple-choice questions, "page turning," and exiting the program. Apple Computer (1986) and Schneideman (1987) have each described various procedures for standardizing program menus. In a similar fashion, standard specifications can be made for identifying multiple-choice alternatives, such as using letters instead of numbers. The keys used to page through a lesson should also be clearly defined and systematized. For example, R may be used to page backwards (review), the space bar to move forward, ESCAPE or Q to exit, and M to return to the main menu. Maintaining consistency of symbols, responses, and format throughout a series of lessons or modules allows the learner to progress through the materials without having to learn a new interface with each new module.

Programming Considerations

The detailed flowchart developed by the lesson designer can also serve as a useful tool for the programmer. After completing the detailed flowchart, line numbers (in BASIC) or labels (in FASCAI or SuperPilot) are assigned to the various frames or segments in the flowchart (see Figure 7). This process leads to the development of a structured program which is easier to revise by grouping related frames into modules. Often a block of frames is assigned a starting number (e.g., 300) with increments large enough to allow adequate coding space between the various blocks. The flowchart has also been used to identify and plan subroutines that can be used for different programs. A library of procedures for creating boxes, borders, prompt lines, help screens, and response checking/editing routines has now been developed. These components are identified on the storyboards and on the detailed flowchart for incorporation into the new program. As different CBI projects are completed, guidelines such as those shown in Table 1 are documented. These guidelines help to ensure that the program can be easily revised if errors are found after the programmer has com-
Table 1

<table>
<thead>
<tr>
<th>Programming guides</th>
</tr>
</thead>
<tbody>
<tr>
<td>Print statements</td>
</tr>
<tr>
<td>Menus</td>
</tr>
<tr>
<td>Questions</td>
</tr>
<tr>
<td>Quit</td>
</tr>
<tr>
<td>Repeat current lesson</td>
</tr>
<tr>
<td>Select menu</td>
</tr>
<tr>
<td>Input</td>
</tr>
<tr>
<td>Data input</td>
</tr>
<tr>
<td>Main menu title</td>
</tr>
</tbody>
</table>

completed the project, or modified for use in a different project.

Summary

This paper has presented a flexible planning process that provides a means of implementing an instructional design on paper as an intermediate step between the design and production of computer-based instruction programs. The model includes a simple syntax for numbering and identifying frames. This syntax can be modified for use with a specific instructional design model (e.g., Merrill, 1983) or for use with an individual designer's own heuristic-based system (Romiszowski, 1981). With this model, the designer has the option of designing screen templates for repetitive frames, or designing individual frames when the content or format changes.

The degree of specificity included in the detailed flowchart is also determined by the individual lesson design. Although the amount of detail included in each step will vary between projects, it is expected that the designer will complete each of the four steps in the model. The purpose of this planning process is to present the designer with a flexible tool that can be used in a variety of CBI projects without requiring unnecessary work.

References


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 (ICA), 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 CAI 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 (Seelbecker, 1989). 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 (Halft, 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, behaviorist 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 IDEF, 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 pre-stored 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 acquisition 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 Briggs'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, Sandora, Land, and Reigeluth have been influential in developing this emerging paradigm (Merrill, Kowalls, & 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, 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, PXE) (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-
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 CAl drill-and-practice programs, but the logical and inferring 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
  - 3
```

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:

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 subtraction 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:

![Diagram of a circuit with components and labels]

A type of skill that an ITS might teach a student from this display would be how to attempt to 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:** (DISORDER A5)

If (And (Causes A5 ES)
(Present ES S5System)
then (present A5 S5System)

**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
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 underevaluated 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 used 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
Needs Assessment and Analysis: Tools for Change

Stephen R. Rodriguez
Project Manager
Center for Instructional Development and Services
Florida State University
Tallahassee, FL 32301-4929

Abstract. The purposes and processes associated with needs assessment, needs analysis, and other front-end activities are discussed. The author notes that the literature presents inconsistent views regarding the purposes of available processes. An organizational model is then presented and used as a basis for categorizing the purposes and utility of assessment and analysis activities. The author distinguishes various kinds of front-end activities from one another and concludes that "holistic needs assessment" is an essential tool for producing meaningful change and achieving lasting organizational success.

Most everyone agrees that needs assessment and associated analysis activities should be completed at the beginning of any contemplated instructional development effort (cf. Andrews & Goodson, 1980; Rossett, 1982). However, extant definitions of assessment and analysis activities are inconsistent (Jonassen & Hannum, 1986; Kaufman, 1982, 1986; Rossett, 1982, 1986).

Is it crucial to distinguish among needs assessment and analysis processes? Some say no, arguing that the central goal of front-end activities is to gather information which will reveal optimal solutions to performance problems (Harmon, 1982; Rossett, 1982, 1986). In contrast, others press for clarification of the operational definitions of various front-end processes. Such clarification, it is proposed, serves to distinguish the purposes of possible front-end processes and enables one to select and complete processes which are appropriate for the targeted planning level (Kaufman, 1982, 1983, 1986).

The divergent views regarding the purposes of front-end processes and the usefulness of differentiating among them imply an important question: Should the scope of front-end activities be limited to solving performance problems, or should one also be prepared to take a larger view of the world in order to conduct an "a priori" assessment of the long-range utility and worth of addressing a given performance discrepancy? Kaufman (1983) suggests that "An over-arching framework which considers and integrates available models, tools, and techniques will be useful for assuring the effectiveness and efficiency of organizational improvement attempts" (p. 3). Kaufman’s view is sound. However, before the suggested integration can occur, further clarification of what various front-end processes entail is required.

Clarification of the purposes and operational definitions of front-end activities will improve our ability to choose and correctly apply them, to communicate our intentions to others, and to move toward synthesis of organizational planning and instructional development models. My purpose here is to contribute to such clarification.

The present discussion, then, considers the purposes and processes associated with holistic needs assessment (Kaufman, 1983) and other front-end activities such as needs analysis, front-end analysis, and task analysis. Linkages between the processes and a conceptual organizational model are presented in an effort to clarify how various processes relate to possible levels of organizational planning. Finally, the optimal context for use of each process is suggested.
Needs Assessment: A State of Confusion?

A common assumption is that needs assessment is prerequisite to development of training. Thus, needs assessment has been variously defined as a means of specifying training requirements (Swierczek & Carmichael, 1985), as a means of determining required skills and knowledge necessary for mastery performance (Deden-Parker, 1980; Harmon, 1982) and as a means of obtaining opinions, ideas, facts, and feelings about a given performance problem (Rossett, 1985). In contrast, others hold that needs assessment may be applied more broadly as a macro-level planning tool used to assess total organizational contributions to society or clients prior to follow-up analysis of related micro-level problems such as deficient performance (Kaufman, 1982, 1983; Kaufman & Stone, 1983; Kinster & Stockton, 1979; Witkin, 1984).

Because existing definitions of needs assessment incorporate diverse frames of reference, confusion results. A definitive frame of reference would allow for more precise specification of the meaning of needs assessment.

The Organizational Elements Model

The Organizational Elements Model (OEM) (Kaufman, 1982, 1983; Kaufman & Stone, 1983) provides a holistic, conceptual overview of organizational resources, activities, accomplishments, and contributions. Clarifying the relationships between means (such as resources and work activities) and end results, the OEM provides a consistent frame of reference for relating needs assessment, analysis processes, management, training, and other interventions (Kaufman, 1983; Kaufman & Stone, 1983; Sample & Kaufman, 1986).

The OEM framework consists of the following five primary components:

- **Inputs** are the existing starting conditions affecting organizational activities. Inputs include physical resources, goals and objectives, laws and policies, and human resources.
- **Processes** are methods of implementing and managing inputs. Also, any job function or activity may be classified as a process.
- **Products** include the internal, end-results accomplished through application of inputs and processes. Examples include finished components (e.g., tended or circuit boards) which are subsequently integrated with other products to form outputs (e.g., automobiles or microcomputers).
- **Outputs** include the services, goods, and aggregated products that an organization delivers to external clients or society.
- **Outcomes** are the impact on clients or society resulting from delivery of outputs. Outcomes are grounded in quality-of-life factors and individuals’ relative levels of self-sufficiency.

The OEM clarifies organizational structure and purpose, illustrates the causal relationships among organizational components, and classifies the components from a holistic perspective. Means include inputs and processes. Internal organizational results are classified as products and outcomes. Outcomes, which represent results external to the organization, are the societal or client-based impact resulting from internal organizational efforts. The distinctions inherent in the OEM establish a basis for defining need and needs assessment.

What Is Needs Assessment?

A need may be defined as a gap between current and desired results. Needs assessment identifies what is and what should be in terms of results and prioritizes the gaps (Kaufman, 1983). Results, as specified in the OEM, include products, outputs, and outcomes. By identifying gaps in outcomes at the beginning of the planning process, one can link instruction and other possible means to the attainment of worthy, long-range results. Kaufman and Stone (1983) provide numerous examples of documented needs for each level of results.

Needs assessments, then, may be conducted from one of two primary perspectives:

1. **A holistic, externally oriented perspective** that looks beyond the organization to consider the outcomes of organizational results (Kaufman, 1977).  
2. **An internal perspective** which accepts the appropriateness of organizational goals and objectives and considers gaps at the product and output levels only.

Holistic needs assessment can be operationally defined as a process including the following major steps: (1) obtaining the commitment and input (i.e., perception data) of representative planning partners; (2) obtaining relevant performance data; (3) analyzing all data; and (4), listing and reconciling differences between current and desired results or outcomes, prioritizing the gaps, and selecting which gaps to close. The steps required to conduct a holistic needs assessment have been explicated by Kaufman and Stone (1983) and Kaufman (1986) and are summarized in Table 1.

Two types of quasi-needs assessments which identify gaps in organizational processes and inputs (means) are
also possible. Quasi-needs assessments are likely to be most useful when they are conducted to ascertain input and process gaps which have been previously linked to documented gaps in results. Needs assessment may be most useful when it utilizes a holistic perspective and focuses first on external organizational results—outcomes. Assessments at other levels can also be useful, depending on the context and purpose of the assessment and the assumptions made about the data that are obtained or collected (Kaufman, 1986; Witkin, 1984).

**Outcome Indicators**

The exact outcomes resulting from delivery of outputs may vary, depending on the organization. For instance, a desirable outcome for a hospital is the return of a patient to good health and full functioning. A desirable outcome for a school is to enable students to attain or exceed self-sufficiency as productive members of society. An outcome for an airline is to transport an individual from one place to another safely, on schedule, with minimal stress, and at acceptable cost.

While the specific outcomes resulting from delivery of outputs vary among organizations, all outcomes ultimately affect people's abilities to be self-sufficient. Thus, one possible measure of organizational success is the extent to which an organization promotes self-sufficiency in members of society.

Self-sufficiency may be defined as the point at which a person is producing as much or more than he or she is consuming, i.e., production ≥ consumption (Kaufman & Stone, 1983). Indicators of a person's relative degree of self-sufficiency include such consequences as employment, financial autonomy, lack of criminal activity, and good credit status.

The holistic approach to needs assessment espoused by Kaufman and others has been criticized as being impractical and excessively difficult to operationalize. This is puzzling since numerous real-world illustrations of using the OEM as a basis for planning have been presented (Kaufman & Stone, 1983). Nonetheless, further specification of outcomes may contribute to the utility of the OEM as a planning framework.

Organizations affect people in at least one of four ways: economically, psychologically, physically, and socio-politically. Examples of these possible categories of "outcome indicators" are presented in Table 2. They are presented as possible bases for measuring organizational outcomes.

**Needs Analysis**

Needs analysis is another process which has been tagged with a variety of definitions. A common view is that needs analysis determines organizational goals and perceived problem areas; considers performance dis-

### Table 1

**Essential Steps in Needs Assessment**

1. Decide to plan using needs assessment data.
2. Select planning level.
3. Identify planning partners.
4. Arrange participation of partners and schedule meetings.
5. Explain OEM and possible needs assessment levels.
6. Obtain acceptance of needs assessment and planning level.
7. Determine gaps to investigate, specify data collection requirements, collect data.
8. Present findings, list documented gaps, complete needs assessment matrix.
9. Discuss, reconcile, and prioritize gaps and derive consensus.
10. List agreed-upon gaps to address.


### Table 2

**Some Self-Sufficiency Indicators**

<table>
<thead>
<tr>
<th>Category</th>
<th>Indicator</th>
</tr>
</thead>
<tbody>
<tr>
<td>Economic</td>
<td>Degree of dependency on others for financial support</td>
</tr>
<tr>
<td></td>
<td>Extent of savings, investments, and real property</td>
</tr>
<tr>
<td></td>
<td>Extent of credit, credit standing</td>
</tr>
<tr>
<td>Psychological</td>
<td>Extent of addictive relationships</td>
</tr>
<tr>
<td></td>
<td>Self-image, state of mental health</td>
</tr>
<tr>
<td>Physical</td>
<td>Degree of fitness, state of physical well-being</td>
</tr>
<tr>
<td></td>
<td>Extent of use of addictive substances</td>
</tr>
<tr>
<td>Socio-political</td>
<td>Extent to which in care or custody of police or governmental agency</td>
</tr>
<tr>
<td></td>
<td>Extent of involvement with governmental rules, policies, and restrictions</td>
</tr>
<tr>
<td></td>
<td>Extent of governmental fines/support</td>
</tr>
</tbody>
</table>
crepancies which hinder attainment of goals; analyzes related job requirements; and assesses existing skills and attitudes of potential trainees in order to derive training requirements. This multi-level "top down" approach was originally presented by McGhee and Thayer (1961) with various phases labelled as organizational, operational, and "man" analyses.

During preliminary organizational analysis, goal attainment is investigated and "organizational climate," senior management's concerns, and current human resources are assessed. During operations analysis, performance discrepancies are described in quantitative and behaviorally specific terms (Zemke & Kramlinger, 1982). During person analysis, existing performance and requirements for improving deficient performance are assessed. This phase is likely to entail testing and performance observation. Collected performance data may be used later when evaluating the effectiveness of the selected intervention (Michalak & Yager, 1979). Moore and Dutton (1978) later based their review of training needs analysis literature on McGhee and Thayer's analytic framework and presented substantive lists of data sources and data collection instruments appropriate for use in the various phases.

Zemke and Kramlinger (1982) also present a multi-level needs analysis model which uses the McGhee and Thayer model as its foundation. In applying this model, one would seek to answer key questions, such as the following: How does current performance impact attainment of organizational goals? In light of organizational goals, which performance discrepancies have the biggest effects? What does an optimally performing person or unit look like? How do the contingencies of reward, punishment, feedback, and other supports affect individuals in the performance environment? This conception of needs analysis articulates important questions regarding internal organizational results and associated processes.

The needs analysis model originally presented by McGhee and Thayer (1961) has been widely cited and utilized in both its original and expanded forms and provides a pragmatic framework for conducting "top-down," internal organizational and performance discrepancy analyses of problems perceived by management (Deden-Parker, 1980; Moore & Dutton, 1978; Scott and Doadrick, 1982; Zemke & Kramlinger, 1982). Needs analysis usually assumes the worthiness of existing organizational goals. It cannot logically occur prior to needs assessment, since the subject and unit of analysis must be determined before analysis may occur. It is useful to the extent that it reveals solutions to performance problems which are linked to gaps in results, especially those at the outcome level.

Front-End Analysis

Front-end analysis (FEA) provides the means for determining causes of performance problems and considering multiple, possible solutions. It is closely related to and often incorporated in needs analysis.

The expression "front-end analysis" is attributed to Joe Harless (1975). He proposes that FEA should reveal answers to the following questions: "What are the symptoms and indicators that a problem exists? What are the performance deficiencies indicated by data? What is the performance problem? What is the relative value of solving that problem?" (p. 29).

Assuming that solving a given problem holds relatively high potential value, Harless recommends investigating all possible (not just probable) causes of the problem. He proposes that performance gaps result from lack of skills and knowledge, practical environmental impediments, or insufficient motivation. The nature of the causes of the problem dictates the solutions.

Romiszowski (1981), concurring with Harless and building on the important work of Mager and Pipe (1970), suggests that possible causes of deficient performance include insufficient skills or knowledge, boredom, inadequate feedback, rewarding or socially reinforcing undesired behavior, or punishing desired behavior. As Harmon (1982) notes, front-end analysis "... will often indicate that changes in physical aspects of the work, the feedback-consequence system, or in supervision will improve performance" (pp. 8-9).

Determining the cause of a performance discrepancy allows one to pursue the appropriate solution. Possible solutions other than training include altering personnel selection criteria, improving supervision, redesigning the job, modifying work conditions, or improving information exchange (Romiszowski, 1981). Providing job aids is often another means of improving job performance.

Successful FEA reveals why actual performance is inconsistent with desired performance, reveals the means by which performance may be improved, and assesses the financial viability of implementing the solution. FEA focuses on internal organizational processes, not on results. Like needs analysis, front-end analysis may be most fruitfully applied when deficient performance is linked to a gap in results, especially at the outcome level.

Needs analysis usually assumes the worthiness of existing organizational goals.
Job Analysis

Job analysis has been defined as "... a process by which jobs are subdivided into elements, such as tasks, through the application of a formalized, systematic procedure for data collection, analysis, and synthesis" (McCormick, 1976, cited in Levine, Ash, Hall, & Sistrunk, 1983, p. 339). Regardless of which job analysis technique is applied, the purpose of such analysis is to produce a validated list of job tasks and to document associated tools, conditions, cues, and standards. Job analysis specifies work activities and can reveal gaps in processes or quasi-needs. Job analysis should not be equated with needs assessment, since it does not lead to determination of gaps in results.

One clear advantage of conducting a job analysis is that it tends to ensure that instruction which is developed includes relevant, job-related tasks and accounts for on-the-job conditions. Job analysis also has utility for designing criterion measures which may be used for purposes of personnel selection, advancement tests, and monitoring the relative adequacy of ongoing job performance. Job analysis may be most useful when it is applied to improve performances that are linked to gaps in results and to attainment of worthy outcomes. Table 3 illustrates the purposes of external, internal, and quasi-needs assessments, needs analysis, front-end analysis, job analysis, and learning analysis relative to the Organizational Elements Model.

Learning Task Analysis

Learning task analysis is an instructional development procedure that occurs after a performance problem has been investigated and instruction has been identified as a viable solution. Assuming that a problem is susceptible to a training solution and that performance requirements are known, learning task analysis should be conducted to determine "... the prerequisites of

Table 3

Relationships of Assessment & Analysis Activities to the OEM

<table>
<thead>
<tr>
<th>Organizational Element</th>
<th>Activity</th>
<th>Purpose</th>
</tr>
</thead>
<tbody>
<tr>
<td>OUTCOMES</td>
<td>External Needs Assessment</td>
<td>Identify gaps in societal/client impact</td>
</tr>
<tr>
<td></td>
<td>Internal Needs Assessment</td>
<td>Identify gaps in deliverable goods and services</td>
</tr>
<tr>
<td></td>
<td>Needs Analysis</td>
<td>Identify causes of perceived or documented gaps in delivered goods and services</td>
</tr>
<tr>
<td>OUTPUTS</td>
<td>Internal Needs Assessment</td>
<td>Identify gaps in en-route results</td>
</tr>
<tr>
<td></td>
<td>Needs Analysis &amp; Front-End Analysis</td>
<td>Identify causes of perceived or documented gaps in en-route results</td>
</tr>
<tr>
<td>PRODUCTS</td>
<td>Quasi-Needs Assessment</td>
<td>Identify performance gaps</td>
</tr>
<tr>
<td></td>
<td>Needs Analysis &amp; Front-End Analysis</td>
<td>Determine causes and solutions of performance problems</td>
</tr>
<tr>
<td></td>
<td>Job Analysis</td>
<td>Document job tasks</td>
</tr>
<tr>
<td></td>
<td>Learning Analysis</td>
<td>Specify prerequisite skills/knowledge required for performance</td>
</tr>
<tr>
<td></td>
<td>Quasi-Needs Assessment</td>
<td>Identify gaps in laws, policies, physical and human resources</td>
</tr>
<tr>
<td></td>
<td>Front-End Analysis</td>
<td>Determine how laws, policies, and work environment affect performance</td>
</tr>
</tbody>
</table>
what is to be learned” (Gagne & Briggs, 1979, p. 105).

Learning analysis entails classifying tasks into one of five categories of learning results: cognitive strategies, intellectual skills, verbal information, motor skills, and attitudes. In addition, procedural components for each task must be analyzed to determine prerequisite skills and knowledge (Gagne, 1985). Once these are determined, instruction can be designed so as to produce the internal conditions in learners required to facilitate attainment of the desired learning results. In training contexts, the utility of instructional analysis techniques presented by Gagne and Briggs (1979), Gagne (1985), Briggs and Wager (1981) and others hinges on whether improving job performance through training will promote attainment of desirable outcomes.

Conclusion

Simplistic or short-sighted approaches are not likely to contribute significantly to true or enduring organizational success or to bring about deep change (Kaufman & English, 1979). The linkage of the Organizational Elements Model with needs assessment and other analysis activities provides the means for attaining meaningful results and establishes a basis for evaluation. By viewing an organization from a holistic perspective, we can consider why an organization exists and plan how to improve people’s lives through meaningful, well-considered interventions.

References


Zemke, R., & Kramlinger, T. (1982). Figure things out: A trainer’s guide to needs and task analysis. Reading, MA: Addison-Wesley.
The Intellectual Content of Instructional Design

Wayne A. Nelson,
Susan Magliaro,
and
Thomas M. Sherman

Division of Curriculum and Instruction
College of Education
Virginia Polytechnic Institute and State University
Blacksburg, Virginia 24061

Abstract. In this paper, instructional design is described as a high-level thinking process. This characterization provides more information regarding the way instructional design is learned and actually practiced in three ways. First, analyzing instructional design from the perspective of cognitive psychology makes possible classification of design models along a continuum from general heuristics to specific algorithms. Second, the literature on planning, schema theory, development of expertise and metacognition supports the notion that instructional design models represent designers' schemata. Third, the development of design schemata can be traced over time to chart the similarities between cognitive abilities of designers. The implications drawn from this analysis can enhance understanding of the design process as well as improve strategies to teach instructional design.

Over the past three decades, instructional design has been characterized in a myriad of models which range from very specific, step-by-step algorithms to general frameworks or heuristics (e.g., Cagne & Briggs, 1979; Sherman, 1980). While these seemingly dichotomous viewpoints appear in various forms in the extant literature (see Andrews & Goodson, 1980, for a review), a reconceptualization of instructional design as a complex intellectual activity similar to other known sophisticated thinking strategies may lead to a more productive understanding of essential processes and strategies. The one ingredient which is consistent in all these models is that their purpose is to guide the decisions that designers must make during the design process.

Extending Simon's (1973) definition of design in general, we propose that instructional design is "a system of productions, in which the elements already evoked from memory and the aspects of the design already arrived at up to any given point would serve as the stimuli to evoke the next set of elements" (p. 190).

From this perspective, the steps and processes of instructional design may be appropriately considered as cognitive tools which must be acquired, developed, selected, and managed through the complex decision-making activities that constitute instructional design. By comparing instructional design to other thinking strategies such as problem solving and decision making, the research and theories of cognitive science can be employed to generate implications and testable hypotheses. Such comparisons may also provide a better understanding of how the cognitive operations which instructional designers use can be learned and developed.

Instructional Design and Complex Thinking Strategies

For many years, those who investigate high-level thinking skills have represented "problems" on a continuum anchored on the one end with "well-defined" and on the other end with "ill-structured" (Greeno, 1976; Reitman, 1965). Well-defined problems

Because designers rarely have clear goals or structured behaviors to solve problems, they typically employ heuristics to guide, organize, or pattern problem solutions.
tend to be solved using an established set of procedures expressed as rules or algorithms. These problems have clear goals and learning to solve them is often a function of time and practice. In many cases, the ultimate ability to solve well-defined problems is functional automaticity in which solution actions operate without conscious attention. In contrast, ill-structured problems have no clear goal or specific solution processes. Consequently, these problems present a different, and perhaps greater, intellectual challenge.

All instructional design problems fall somewhere along this continuum of well-defined to ill-structured (see Figure 1); however, “design” problems by and large fall toward the ill-structured anchor. Because designers rarely have clear goals or structured behaviors to solve problems, they typically employ heuristics to guide, organize, or pattern problem solutions. Similarly, scientists and artists, among others, lack specific goals and procedures to produce products. As a result, they employ heuristics, such as the scientific method, problem solving, creativity, and decision making, which provide a general model of the processes they use. Whether these heuristics operate like algorithms or serve as a theoretical schema for systematic problem exploration has been the subject of controversy (e.g., Carroll, Thomas, & Malhotra, 1980). Nonetheless, within the domain of instructional design, little attention has been given to identifying and explaining the intellectual activities associated with designing instruction.

**Instructional Design Thinking**

In many respects, the structure or components of instructional design are similar to those of general problem solving (see Polya, 1957). Instructional design begins with understanding the problem by establishing goals, analyzing needs and learner characteristics, and identifying the scope and content of the subject. Once the “problem space” has been defined, the designer breaks the problem into a series of subgoals that, if reached, will solve the problem. These component activities can include potential actions such as identifying and sequencing objectives, devising activities, choosing materials, and planning evaluations. In this way, an ill-structured, global design problem with broad definitions becomes more structured and specific (Simon, 1973).

In reviewing the literature, we found that design models, like problems, could be distributed along a continuum ranging from broadly defined, general heuristics to specific step-by-step algorithms. General models (e.g., Wildman, 1980) provide an overall, yet abstract, sequence of mental processes or strategies. On the other hand, algorithmic models (e.g., Annarino, 1983) are step-by-step procedures that are developed for a specific content area and outline each incremental design activity in a very concrete manner. These examples illustrate relatively well-defined points on this instructional design continuum. Others, such as Scandura (1983) and Land (1976) present positions which differ in specificity and structure and may be less reliably placed on such a continuum.

On the surface, design models give the designer a set of actions for planning by providing an external framework to guide the design process. But at a deeper level, design models represent a mental framework or schema of a designer’s internal organization of knowledge and information about instructional design. Consequently, in addition to depicting instructional design as a product-oriented activity, instructional design models should also be considered to represent intellectual processes. These cognitive activities of designing instruction are epitomized by the interactions between the complexity of the designer’s cognitive structures, the instructional design task, and the environment (see Figure 2).

**Theoretical Background**

Though little research on the thought processes of instructional designers exists, there is considerable evidence from cognitive psychology which supports the importance of the relationship between cognitive activities and behavioral outcomes (Ericsson & Simon, 1984). In a study of instructional designers’ thought processes, Nelson (1987) compared the design models and knowledge of experts and novices. Not surprisingly, differences between experts and novices mirrored the findings from previous expert/novice comparison studies in such domains as

**Figure 1. A continuum illustrating the relative characteristics of the structure of specific vs. general instructional designs.**

<table>
<thead>
<tr>
<th>Well Structured, Specific, Algorithmic</th>
<th>Moderate Structure</th>
<th>Ill Structured, General, Heuristic</th>
</tr>
</thead>
<tbody>
<tr>
<td>Step-by-step procedure to produce an instructional product</td>
<td>General rules and procedures with specific examples</td>
<td>General concepts which follow a “design” sequence</td>
</tr>
</tbody>
</table>
chess (e.g., Chase & Simon, 1973), electronics (e.g., Egan & Schwartz, 1979), and physics (e.g., Larkin, McDermott, Simon, & Simon, 1980). That is, in comparison to novices, experts’ knowledge structures are more highly organized and well-integrated, and experts are able to respond to and complete tasks more quickly and systematically. The literature on planning, schema theory, development of expertise, and metacognition provides an extensive analysis of the mental activities associated with complex intellectual tasks which may also add to our understanding of the thought processes of expert instructional designers.

Planning

Using their theory of human problem solving, Newell and Simon (1972) compared planning to the use of an algorithm in solving a problem. Specifically, plan construction was viewed as an auxiliary activity that aids in the solution of a problem by guiding the action in the problem space. Newell and Simon proposed that a plan is generated by: (1) abstracting details from original objects and operators; (2) forming a similar abstract problem; (3) solving this abstract problem and using the solution to provide a plan for solving the original problem; and (4) translating the plan back to the original problem space for execution.

This planning process has been modeled on computers by several researchers (Newell & Simon, 1972; Sacerdote, 1974). Called automatic planning systems, these models operate by a process of successive refinement in which a solution is generated, compared to a standard, modified, compared again, and so on until a good (or acceptable) outcome is achieved. Though Fikes (1977) reported several methods used to represent knowledge in these systems, typically a goal state is specified and the planner matches the goal to an action that will achieve some portion of the goal. The goal is then reconsidered in light of that new action. This kind of planning process is called “top-down” because solutions are progressively developed by moving from general to specific states of abstraction.

Hayes-Roth and Hayes-Roth (1979) propose an alternative model of planning as an information-enriching activity which involves sorting and comparing multiple alternatives. The essential action is relating generated alternatives to desired goal states and judging their adequacy. Computer planning programs are often effective because of their capability to quickly and tirelessly rate the salience of information or cues and to compare alternatives; consequently, efficiency is not a problem. Humans, however, have neither the memory power nor the immediate access characteristics of computers. Thus, conceptual schemata or knowledge structures become critical components of planning in order to manage and control cognitive operations with any precision.

For example, ordinarily designers organize a design task with either a general heuristic or an algorithm. The conceptual structure serves as a means to extract needed information and relate the information to intended outcomes. Through an iterative process of

![Figure 2. A representation of the interaction between the designer and the design environment.](Image)

1988, VOL. 11, NO. 1
Despite having no content expertise, the designer has a set of representations based on a "design model" which can guide the development of an effective training program.

The notion of schemata as basic, stereotypical descriptions of general knowledge was first proposed by Bartlett (1932). More recently, Alba and Hasher (1983) have elaborated the definition of a schema as a framework which "selects and actively modifies experiences in order to arrive at a coherent, unified, expectation-confirming and knowledge-consistent representation of an experience" (p. 203).

Schank and Abelson described schemata or scripts as stereotypical representations of common events, activities, or settings. Scripts may contain general information about a setting and specific information about the activities which may occur in that setting. Schank and Abelson (1977) have used scripts as the central organizational element in their research on computer understanding of natural human language.

Using this script theory, Bower, Black, and Turner (1979) studied understanding and memory of narrative text passages. Their findings indicated that subjects generally agreed on the conventional roles, props, and sequences of standard activities such as eating in a restaurant or going to a concert. Moreover, comprehension and recall of narrative passages was shown to be related to the degree to which a passage resembled an existing script. Thus, these scripts exert a powerful influence on comprehension. In effect, comprehension is a constructive activity rather than a background task of "taking meaning off a page."

Development of Expertise

Piaget (1970) proposed that cognitive structures or schemata change as individuals encounter different situations and tasks. He believed that schema development follows an orderly evolution through various stages which represent qualitative change to progressively more sophisticated levels of reasoning. These changes, rather than being isolated, permeate and reorganize many different schemata. The individual plays an important role in this process by comparing new and old schemata, recognizing changes based on new experience, and actively reorganizing schemata to promote new understanding.

In a study by Pitt (1983), problem-solving schemata were shown to be essentially the same for adolescents and adults. However, significant developmental differences between age groups...
Despite having no content expertise, the designer has a set of representations based on a “design model” which can guide the development of an effective training program.

Consider the position of instructional designers developing a training program on tasks which are unfamiliar to them. Despite having no content expertise, the designer has a set of representations based on a “design model” which can guide the development of an effective training program. By following this script of cognitive behavior, the designer can “think through” design procedures and initiate design activities consistent with the design model.

Jeffries, Turner, Polson, and Atwood (1981) examined the role of schemata in the design of computer software. They contend that a general design schema for software designers exists which contains abstract knowledge about the overall structure of a design as well as the processes involved in generating that design. They assume that, at least for expert software designers, design schemata would be similar, although individual differences between experts could exist because of variations in software design experience. These schemata appear very similar to what are called design models. Based on schema theory, we can expect the complexity and generality of designers’ schemata to influence design conceptualizations, activities, and outcomes. We can also expect these schemata to change with experience, time, and reflection.

Development of Expertise

Piaget (1970) proposed that cognitive structures or schemata change as individuals encounter different situations and tasks. He believed that schema development follows an orderly evolution through various stages which represent qualitative change to progressively more sophisticated levels of reasoning. These changes, rather than being isolated, permeate and reorganize many different schemata. The individual plays an important role in this process by comparing new and old schemata, recognizing changes based on new experience, and actively reorganizing schemata to promote new understanding.

In a study by Pitt (1983), problem-solving schemata were shown to be essentially the same for adolescents and adults. However, significant developmental differences between age groups...
were found in how problems were defined and in the use of problem-solving strategies. Adults were shown to use qualitatively different subroutines and deductive strategies for problem solving. For example, Pit gave a chemistry problem to 10th graders and college juniors who had no significant background in chemistry. By measuring the subroutines of the solution process for each subject, Pit showed that, although the 10th-grade students followed the same general heuristics and strategies as the older group, they "defined problems inadequately and generated ill-formed, inaccurate hypotheses," as well as possessed "limited ability to coordinate information or operations" (p. 547).

Other studies have also demonstrated significant differences in problem-solving abilities between experts and novices in such domains as chess (Chase & Simon, 1973), physics (Chi, Glaser, & Rees, 1982; Larkin, McDermott, Simon, & Simon, 1980), political cognition (Fiske, Kinder, & Larter, 1983), and computer programming (McKeishen, Reitman, Ruetter, & Hittle, 1981). In general, these studies show differences in task performance and problem representation based on experience in the problem domain. Experts tend to "chunk" or organize information into more highly structured patterns and to complete the task more quickly than novices. Experts also appear to represent problems differently than novices because of their superior ability to recognize patterns, infer relationships, disregard irrelevant information, and recall similar problems from past experience.

Differences between expert and novice instructional designers are likely to be similar. For example, novice designers are more likely than experts to use design models at a surface level. As a result, novices may miss important, though subtle, characteristics of performance and the developmental experiences which lead to satisfactory performance. Training programs based solely on the behaviors of established masters could ignore the similarities between components of the criterion task as well as optimum training procedures to teach trainees when to generalize and discriminate the application of learned competencies. With experience and reflection, expert instructional designers should represent design problems more accurately and in a more organized manner. Thus, expert instructional designers' schemata should be governed by qualitatively superior intellectual strategies which make possible a better understanding of problems and the process by which they may be solved. Cognitive scientists have named these higher order cognitive skills metacognition.

Metacognition

The cognitive functions of experts are likely to proceed in a hierarchical manner with superordinate processes controlling more basic subordinate processes. Kagan (1976) has identified some of these higher level metacognitive or executive functions, to include: (1) recognition of the nature of a problem and adjustment of effort to task difficulty; (2) flexibility with respect to alternatives; (3) activation of organization and rehearsal strategies; (4) control of distraction and anxiety; and (5) faith in the power of thought.

Implications

The apparent similarity between the characteristics of high-level modes of thought and those required for instructional design raises implications for training and future research. First, it appears that an instructional design model represents the design schema of its user. In part, this explains the proliferation of design models that has occurred (e.g., Andrews & Goodson, 1980). Each model is in effect, a schema of the specific actions required to solve the design problem it addresses. What is important, however, is that these models must match the users' cognitive abilities. Just as there are different stages of cognitive development, similar levels can be identified with respect to models of instructional design. Thus, these models tend to differ on levels of specificity, from elaborate and algorithmic to simple and generic. In practice, users must find or generate a model that matches their existing cognitive abilities.

Just as there are different stages of cognitive development, similar levels can be identified with respect to models of instruction.

The role of executive functions was investigated in a study by Hayes-Roth (1980) in which the predictions made by the opportunistic model of planning (Hayes-Roth and Hayes-Roth, 1979) were tested for an errand-planning problem. The results of this study showed that people can acquire new executive strategies by instruction or experience, and adjust old or adopt new executive strategies according to the problem characteristics. There was also an indication that "knowledge of
An interesting paradox is that, as design models become less specific, designers need more highly developed and sophisticated schemata to implement the subprocesses that are subsumed under the various stages of design. Like expert chess players, expert instructional designers must have a broad knowledge base or vocabulary of moves which is well developed and categorized in order to process information efficiently and organize it into large chunks. This intellectual structure allows designers to retrieve and use their schemata more effectively as they receive new information from the problem space.

Second, because instructional design involves systematic thinking, clearly stated purposes must follow an orderly structure to their conclusions. As a design project progresses and becomes more well defined, thinking shifts from broad considerations such as needs analysis, to more specific issues such as specification of individual lesson objectives. This is not to say that instructional design is a top-down process, however. One of the features of the planning model proposed by Hayes-Roth and Hayes-Roth (1979) is the flexibility to move from abstract to specific levels and back again. Decisions made in the initial stages of planning can affect subsequent stages, and the designer must be aware of these possibilities. This aspect of planning is controlled by the executive strategies which are elements of the designer's metacognitive structures.

Third, a broad knowledge base is needed. Boery, et al. (1981) have identified domains of knowledge which they consider essential for instructional designers. These include systematic planning, learning theory, instructional theory, and educational measurement. Basic skills in interpersonal communication, content analysis, task analysis, learner analysis, specification of objectives, and interpretation of data were also listed as important. Integration and restructuring of designers' schemata are necessary for the knowledge and skills to be used effectively, however. As Piaget's theory indicates, the process of cognitive development is continuous. As one develops expertise in a domain, skills are refined and knowledge broadened as designers gain experience working through successive projects. Again, a metacognitive awareness of designers' knowledge can assist in this development. If deficiencies are known, steps can be taken to acquire the needed information from sources outside of the designers' memory.

Finally, creativity and flexibility appear to be important, if not necessary, components of high-level thinking. Sachs (1981) has claimed that, "It is the developer's judgment, sensitivity, and inventiveness that leads to success" (p. 7). Instructional design models provide a proper framework for planning, management, and organization of information. The models often fail, however, to specify when a step is completed, or what to do to attack problems. Rote adherence to a model does not guarantee an optimum product. Many decisions are influenced by outside constraints and interactions between people and events. Therefore, the ability to be flexible and creative in response to difficulties should be of concern. For example, it appears that attitudes can be as important as the methods used. Anderson (1980) reported research suggesting that if people imagine themselves as creative, this can lead to an increase in the creative ideas they produce.

Summary

We have characterized instructional design as an intellectual activity. As such, instructional design can be understood with reference to the way individuals think, develop design schemata, and control and use specific and general design knowledge.

By examining the intellectual properties of instructional design strategies, designers can use models more efficiently and better understand the thinking required to design instruction.

References

Anastario, A. A. (1983). The teaching-learning...


Allison Rossett
Book Review Editor
Professor of Educational Technology
San Diego State University
San Diego, CA 92182-0311


I've been looking forward to this book since I participated in one of Allison Rossett's workshops a few years ago. It's been worth the wait. I can't imagine anyone who is, or plans to be, involved in any aspect of front-end analysis not finding Training Needs Assessment a real "page-turner." It's very well organized and just packed with useful and often humorous anecdotes and examples. But what really provides the momentum of Training Needs Assessment is the presence in every chapter of the author's unique voice. Rossett's enthusiasm for her subject is infectious, and her respect for others who have contributed to the body of knowledge around her subject is obvious. She takes great care to acknowledge the origins of ideas and concepts we have come to associate with training needs assessment. (As I was reading, I found myself saying things like: "Gee, I didn't know we got that idea from Dewey," or "I didn't know we owed Bandura for that one.")

What really "drives" this book, however, is Rossett's keen interest in the application of the body of knowledge to real problems. Every chapter is enriched by examples taken from her own consulting experiences or from those of her graduate students. There seems to be no limit to the variety of examples Rossett uses to illustrate the concepts in this book or their application: banking, telephony, cockroach control, making pizzas, selling programs at the ball park. . . .

The book is divided into four major sections: Introduction, Techniques, Tools, and Conclusion. In the Introduction section, Rossett establishes the practical context of the book by presenting and illustrating three kinds of situations which typically initiate training needs assessment (TNA) activities:

- **Performance problems**, where employees ought to know what to do, but aren't doing it.
- **New systems or technologies**, where employees don't know what to do because they've never been taught.
- **Habitual or automatic training**, where no particular problem exists, but training on a subject has become traditional.

It's within this context that Rossett introduces her central concept, Purpose-Based Assessment, as "an inclusive way of thinking about" any type of training need. Rossett describes five purposes which can be fulfilled by Purpose-Based Assessment:

1. To define *optimal*, or what should be happening.  
2. To define *actual*, or what is happening.  
3. To describe the feelings of employees or significant others around a situation or its proposed solutions.  
4. To identify probable *causes* of problems.  
5. To identify *solutions*.

Building on examples presented earlier, Rossett shows how Purpose-Based Assessment can be applied to a variety of situations. For example, a needs assessment initiated by a performance problem must satisfy all five purposes, but when a new system or technology initiates the need, the assessment tends to focus on three of the five purposes: optimal, feelings, and solutions. This useful way of thinking about "what"—what situations have triggered a particular TNA—and also what purposes must be fulfilled in its assessment—not only helps analysts clarify their goals, but also provides direction for their thoughts about "how" the TNA can be achieved most efficiently.

"How" is the subject of the balance of Training Needs Assessment: Techniques for achieving the purposes of the TNA: Tools for implementing the techniques; and, in the Conclusion section, planning and reporting the results of the TNA.

In the second section of the book, Rossett cites four major training needs assessment techniques:

1. **Extent Data Analysis**, which involves drawing inferences about performance from data which describe its results.
2. **Training Needs Assessment itself**, which Rossett describes as the technique of gathering opinions about key purposes.
3. **Subject Matter Analysis**, the description and organization of invisible bodies of knowledge.
4. **Task Analysis**, the description and organization of observable behaviors.

Rossett deals with the first three of these techniques in detail here, but refers us to Ken Carlisle's recent book, Analyzing Jobs and Tasks (1986), for information on the Task Analysis technique.

Rossett has organized her discussion of the techniques—and thus has simplified their comparison—by pre-
senting every chapter on techniques in six parts:

- **Purpose**, which includes an explanation of the relationship of the technique to the five purposes of Purpose-Based Assessment.
- **Description**, in which the features and benefits of the technique are discussed.
- **When** to use the technique.
- **How** to use it, a step-by-step application of the technique.
- A complete **Example** of the technique in practice.

The chapter on Training Needs Assessment includes Rossett’s Typology of Questions. This Typology represents six categories of question types, each of which relates to at least one of the purposes of Purpose-Based Assessment. The Typology provides a basis for making informed decisions about techniques and tools for a particular assessment. It also provides a system for tracking, and when necessary, for altering the direction of assessment activities. Application of the Typology is supported by useful job aids and a rich variety of examples.

The third section of the book, on TNA tools, focuses on these four:

1. **Interviewing**
2. **Observing**
3. **Working with Groups**
4. **Writing Questionnaires and Surveys**

As she does in the Techniques section, Rossett also organizes her discussion of TNA Tools in a consistent pattern:

- **Introduction**, a general description of the tool.
- **Purposes**, which relate back to the Purpose-Based Assessment concept.
- **The Power and Challenges** of the tool.
- **Steps** for using the tool.

This section is packed with examples which Rossett often uses to introduce the chapters, and then embellishes to illustrate key characteristics of the tool as she develops her subject. There are also some very practical checklists which can be used to guide the planning, development, and application of the tools.

The Conclusion of Training Needs Assessment deals with “Planning” and “Reporting Results.” In the chapter on “Planning,” Rossett provides useful guidelines for assessing the “context” of the TNA, and in doing so, reinforces the real-world focus of the book. She illustrates the guidelines with examples embedded in a general review of all the previous chapters. Rossett wraps up this chapter with descriptions of the planning process for two very different TNA situations, which further demonstrate the flexibility of Purpose-Based Assessment.

“Reporting Results,” a subject often overlooked, is thoroughly discussed in the final chapter of the book. Rossett deals with the importance of communicating results, both during and at the end of the needs assessment process, and of considering purposes, audience, and institutional time constraints. This chapter also contains an interesting section on using computers and software both to generate tools and to analyze and report results.

Using the key concept of Purpose-Based Assessment as an organizational vehicle, Training Needs Assessment provides a comprehensive view of every major type of activity that front-end analysis can involve: from initiation, through goal setting, planning, selecting techniques and developing data collection tools, to reporting results.

I’m going to buy copies of Training Needs Assessment for the people who work with me. I would recommend that it be added to your library if you are involved in any aspect of front-end analysis. It gave me techniques and tools that I was able to use immediately. It also sent me to the library to check out some of my “idea heritage” as a training needs analyst. And it helped put some of the “trials” of doing TNA into perspective by letting me laugh at myself and my institution, while at the same time, giving me a sense of being an active participant in an emerging profession—Reviewed by Judith Fidler, Manager of Training & Development, British Columbia Telephone Company.

This assessment of the status of research on instructional development (ID) discusses such conceptual problems as the confusion between ID and instructional design, reviews previous research, and proposes a framework for future ID research. Decision-oriented issues discussed within this framework include: (a) administrative and policy issues of ID agencies; (b) internal organization and management of ID teams; (c) interaction with clients; (d) social and political relationships with supra-systems; and (e) optimization of ID procedures such as model development, needs analysis, objectives specification, prototype construction, formative evaluation, summative evaluation, and implementation.

Conclusion-oriented issues, a second broad category within the framework, are discussed in the context of the definition and effectiveness of ID. The behavior of participants in the ID process and the effects of ID products on their intended audiences are also identified as major classes of phenomena to be studied, and direct, indirect, and simulated observation are suggested as possible research methodologies. (60 references.) Microfiche 78 cents, paper copy $1.85 plus shipping, as document ED 285 551.


The purpose of this study was to determine whether different types of practice strategies are equally effective in facilitating encoding strategies and in subsequent information acquisition and retrieval by students identified as possessing high, medium, and low levels of prior knowledge in a content area. Within the context of the encoding-specificity principle, the effectiveness of verbal and visual testing of information acquired from visually complemented instruction was also investigated.

Subjects were 240 graduate and undergraduate students who were randomly assigned to eight treatment groups which determined the level of practice received and the testing mode (verbal or visual). Each student received a pretest, participated in the instructional presentation, and then received a battery of tests designed to measure various educational objectives. Results indicated that practice strategies: (a) can be effective instructional variables in improving student achievement; (b) function differentially in promoting student achievement of different educational objectives; and (c) can function to reduce differences among students possessing different levels of prior knowledge. It was also found that visual tests appear to be more effective than verbal tests in assessing information communicated by visualized instruction.

Supplemental materials include sample test questions, statistical data, and 54 references. Microfiche 78 cents, paper copy $3.70 plus shipping, as document ED 285 553.


This review of the research on aptitude-treatment interaction (ATI) discusses four methods or assumptions that might explain why ATI research has not generated the anticipated empirical support: (a) the lack of a theoretical base; (b) disagreement over what a given aptitude means and how it should be measured; (c) difficulties in defining instructional methods as treatments; and (d) the inability to generalize from context-specific results. Increased recognition of the role of prior knowledge as an aptitude variable and the more precise definition of instruction treatments are noted as recent, promising approaches to this research.

Scenarios for the future are suggested which pertain both specifically to ATI research and to the more general implications of such research for instructional design. The importance of adapting instruction to individual learner characteristics such as interest and cognitive style is emphasized throughout. (14 references.) Microfiche 78 cents, paper copy $1.85 plus shipping, as document ED 285 552.

Psychotechnology as Instructional Technology: Systems for a Deliberate Change in Consciousness, David G. Gueulette and Connie Hanson. Paper presented at the annual convention of the Association for Educational Communications and Technology, Research and Theory Division, Atlanta, 1987. 18pp.

Supplemental materials include sample test questions, statistical data, and 94 references. Microfiche 78 cents, paper copy $3.70 plus shipping, as document ED 285 553.
This discussion of the importance to education of psychotechnologies such as biofeedback, meditation, and guided imagery focuses on the potential of such techniques to expand human learning capabilities and consciousness. The work of many theorists and researchers in the fields of education, physiology, and psychology is reviewed, citing evidence that psychotechnology can: (a) augment instructional design models; (b) generate extraordinary learning outcomes; (c) provide additional cognitive strategies to improve one's ability to learn; and (d) augur a future trend in the definition of instructional technology. (47 references.) Microfiche 78 cents, paper copy $1.85 plus shipping, as document ED 285 539.

The Effects of Locus of Instructional Control and Practice on Learning from Interactive Video, Michael J. Hannatin and Maryanne E. Colamaino. Paper presented at the annual convention of the Association for Educational Communications and Technology, Research and Theory Division, Atlanta, 1987. 16pp.

The effects of various interactive video instructional control options and practice on learning were examined in this study. The interactive video lesson was a 30-minute videotape designed to introduce cardiopulmonary resuscitation (CPR). Subjects were 48 graduate and undergraduate volunteers, none of whom had prior experience with CPR or interactive video. Students were randomly assigned to one of three instructional treatments with differing versions of locus of instructional control: (a) designer imposed, following a predetermined path through the lesson dependent on responses to embedded practice questions; (b) learner selected, allowing individual control of decisions at certain points; or (c) linear, with no options for control or imposed decisions for remediation or question repetition. A posttest was administered to assess the learning of facts, procedures, and problem-solving skills. Both the designer-imposed and learner-selected groups performed better than the linear group, and scores on practiced items were higher than non-practiced items for each type of learning. These effects were greatest for factual learning and least influential for procedural learning.

Supplemental materials include 41 references, sample practice questions, and a graph showing the interaction between practice and type of learning. Microfiche 78 cents, paper copy $1.85 plus shipping, as document ED 285 541.


The goal of the programmatic research program for the Minnesota Adaptive Instructional System (MAIS), an intelligent computer-assisted instruction system, is to empirically investigate generalizable instructional variables and conditions that improve learning through the use of adaptive instructional strategies. Research has been initiated in the integration of individual difference variables within the student model by extending the learner assessment process to include cognitive, affective, and memory models, and instruction variables associated with the learning conditions of verbal information and cognitive strategies have been tested. Two important features of the current version of MAIS are that it distinguishes between individualized instruction and self-instructional teaching, and it employs a cognitive psychology approach to the selection of instructional strategies.

Within the macro or curriculum level of the MIS, variables are defined that relate directly to the concepts of memory and cognition, while at the micro or instructional level, variables are defined that relate to the concept of learning. These two levels interact in an iterative fashion such that the initial conditions of instruction established by the expert tutor model in the macro level adapts at the micro level according to learner progress and needs in learning. Descriptions of the nine instructional variables that form the possible meta-instructional strategies and a discussion of continuing research directions conclude this paper. (13 references.) Microfiche 78 cents, paper copy $1.85 plus shipping, as document ED 285 528.

An Historical Analysis of Form, Style, and Instructional Design in Teaching Films, Barbara Erdman. Paper presented at the annual convention of the Association for Educational Communications and Technology, Research and Theory Division, Atlanta, 1987. 9pp.

A study is proposed to address five questions about the interrelationship of instructional intent and film form and style: (a) What are the stylistic elements within the teaching film, and how are these elements used to present epistemological content? (b) Does the instructional component of the teaching film take on a form or stylistic function in the film? (c) In what way does the lesson structure the teaching film? (d) Is the form of the filmed lesson determined by design practices prescribed by formal theories of learning and instructional design research? and (e) in what ways do educational films specifically structure the learning experience?

The proposed study will perform a systematic analysis of a historical sampling of educational films produced between 1930 and 1970 to obtain a detailed picture of the phenomenon of educational film, and to develop a model of the teaching film of this period. The study will be based on the assumptions that educational films existed in an environment dominated by Hollywood films, and that producers of educational films were aware of work being done by others in the field. (11 references.) Microfiche 78 cents, printed copy $1.85 plus shipping, as document ED 285 535.

The above documents may be ordered from the ERIC Document Reproduction Service (EDRS), 3500 Wheeler Ave., Alexandria, VA 22304-5110. Please order by ED number, indicate the format desired (microfiche or paper copy), and include payment for the price listed plus shipping. For Visa or MasterCard orders or information on shipping costs, call EDRS at 1-800-227-3742. Inquiries about ERIC may be addressed to the ERIC Clearinghouse on Information Resources, 630 Huntington Hall, Syracuse University, Syracuse, NY 13244-2340 (315-423-3640).