Training for Reflective Expertise: A Four-Component Instructional Design Model for Complex Cognitive Skills

□ Jeroen J. G. van Merriënboer
Otto Jelsma
Fred G. W. C. Paas

This article presents a four-component instructional design model for the training of complex cognitive skills. In the analysis phase, the skill is decomposed into a set of recurrent skills that remain consistent over problem situations and a set of nonrecurrent skills that require variable performance over situations. In the design phase, two components relate to the design of practice; they pertain to the conditions under which practice leads either to rule automation during the performance of recurrent skills or to schema acquisition during the performance of nonrecurrent skills. The other two components relate to the design of information presentation; they pertain to the presentation of information that supports the performance of either recurrent or nonrecurrent skills. The basic prediction of the model is that its application leads to "reflective expertise" and increased performance on transfer tasks. Applications of the model that support this prediction are briefly discussed for the training of fault management in process industry, computer programming, and statistical analysis.

□ This article presents an instructional design model for training complex cognitive skills or high-performance skills. The model is related to recent psychological theories on learning and information processing. In addition, applications of the model in three different domains are presented. For the purpose of this article, we adopt Schneider's (1985) definition of high-performance skills. He defined this type of skill as having at least the following three characteristics: First, the trainee must expend considerable time and effort to acquire an acceptable mastery level. Second, a substantial number of individuals will fail to develop proficiency. Third, there will be qualitative differences in performance between novices and experts. Examples of complex cognitive skills are fault-management skills in process industry, computer programming skills, statistical analysis skills, air traffic control skills, production and inventory management skills, and military air weapons control skills.

Many instructional design models offer guidelines for training one-dimensional skills (i.e., skills that aim at homogeneous sets of learning objectives) required to perform, for instance, recall tasks, classification tasks, procedural tasks, and causal reasoning tasks (see Reigeluth, 1983a). However, only a few models provide guidelines for training multidimensional, complex cognitive skills that involve several component skills (i.e., skills that aim at integrated sets of various objectives; Fabiani, Buckley, Gratton, Coles, & Donchin, 1989; Fisk & Gallini, 1989; Myers & Fisk, 1987; Schneider, 1985; Tennyson & Rasch, 1988). Actually, this is not surprising because there are still many intriguing questions regarding the underlying cognitive processes in learning complex cognitive skills. For instance, there is the vivid debate on the long-lasting problem concerning the relative importance of domain-independent versus domain-specific skills (Perkins & Salomon, 1989). Related to this issue, there is the discussion regarding the trainability of domain-independent skills and the problem of how to obtain transfer of acquired skills and knowledge to a different set of task demands in new situations.

Today, the lack of detailed instructional design models for training complex cognitive skills may be considered to be a serious omission in the area of instructional technology. This is particularly true because, in a society that is characterized by fast technological changes, many routine tasks are taken over by machines and new tasks that require more and more flexible problem-solving behavior rapidly emerge. Consequently, there is an increasing need for effective instructional guidelines for the training of intensive problem-solving cognitive skills.

The goal of training programs for this type of skill cannot be restricted to obtaining a level of proficiency at which the skills can be performed quickly, fluently, effortlessly, and relatively automatically (rule-based behavior). Instead, in instructional settings for the training of high-performance skills, one should aim at *reflective expertise* (Olsen & Rasmussen, 1989). This kind of expertise entails the ability to perform familiar aspects of a task highly automatically, so that processing resources become available that may be used to interpret new, unfamiliar aspects of the problem situation in terms of generalized knowledge about the task. This combination of rule-based and knowledge-based behavior implies transfer of acquired skills and knowledge (Thorndyke & Hayes-Roth, 1979). In this article, it is argued that the nature of reflective expertise itself offers various points of contact for the development of instructional design models for the training of complex cognitive skills.

In the second section of this article, the outline of a four-component instructional design model (4C model) and its predictions about transfer are presented. The 4C model can be seen either as a generalization of van Merriënboer's model (1990a, 1990b) for training computer programming skills or as an extension of the ADAPT (Apply Delayed Automation for Positive Transfer) design model (Jelsma, van Merriënboer, & Bijlstra, 1990; Pieters, Jelsma, & van Merriënboer, 1987) with an analysis phase. In addition, starting from selected instructional tactics, the 4C model explicitly pays attention to the composition of adequate training strategies that can be applied in interactive learning environments.

In the third section of the article, application of the 4C model is illustrated by examples of instruction for fault-management skills in process industry, computer programming skills, and statistical analysis skills. Finally, a discussion of the model is presented, and its implications for future research in both cognitive science and instructional science are considered.

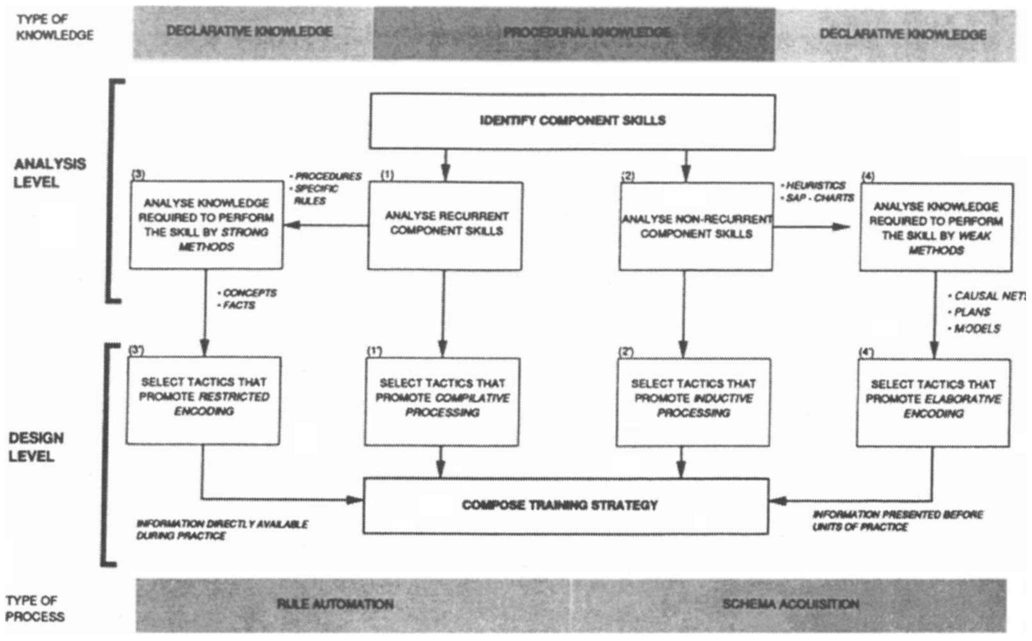
THE FOUR-COMPONENT INSTRUCTIONAL DESIGN MODEL

Figure 1 illustrates the basic elements of the 4C model. Application of the model requires a systematic approach. The process of building a training strategy for complex cognitive skills essentially passes through two general phases: analysis and design.

The Analysis Phase

The main goal of the analysis phase is to describe all component skills that make up the complex cognitive skill. First, the component skills are identified and classified as either recurrent skills, which remain consistent over various problem situations, or nonrecurrent skills, which require a variable performance over situations. Recurrent skills may be further classified as either procedural or nonprocedural. Then, the component skills are described by means of skills and knowledge analyses.

FIGURE 1 ☐ An Outline of the Four-Component Instructional Design Model


TYPE OF KNOWLEDGE
DECLARATIVE KNOWLEDGE
PROCEDURAL KNOWLEDGE
DECLARATIVE KNOWLEDGE
ANALYSIS LEVEL
IDENTIFY COMPONENT SKILLS
(3) ANALYSE KNOWLEDGE REQUIRED TO PERFORM THE SKILL BY STRONG METHODS
• PROCEDURES • SPECIFIC RULES
(1) ANALYSE RECURRENT COMPONENT SKILLS
(2) ANALYSE NON-RECURRENT COMPONENT SKILLS
• HEURISTICS • SAP - CHARTS
(4) ANALYSE KNOWLEDGE REQUIRED TO PERFORM THE SKILL BY WEAK METHODS
• CONCEPTS • FACTS
• CAUSAL NET
• PLANS
• MODELS
DESIGN LEVEL
(3') SELECT TACTICS THAT PROMOTE RESTRICTED ENCODING
(1') SELECT TACTICS THAT PROMOTE COMPILATIVE PROCESSING
(2') SELECT TACTICS THAT PROMOTE INDUCTIVE PROCESSING
(4') SELECT TACTICS THAT PROMOTE ELABORATIVE ENCODING
COMPOSE TRAINING STRATEGY
INFORMATION DIRECTLY AVAILABLE DURING PRACTICE
INFORMATION PRESENTED BEFORE UNITS OF PRACTICE
TYPE OF PROCESS
RULE AUTOMATION
SCHEMA ACQUISITION

Identification of Component Skills

The first step is to decompose a complex cognitive skill into two sets of component skills. One set of component skills is formed by recurrent skills; essentially, these skills are performed in a similar way over various problem situations. The other set is formed by nonrecurrent skills; performance of these skills may vary considerably over various problem situations. Recurrent skills, which are sometimes called consistent component skills or closed skills (Schneider & Fisk, 1982), are characterized by the fact that invariant rules or procedures, invariant components of processing, or invariant sequences of information-processing components are used in successful performance (Fisk & Gallini, 1989). For instance, in supervising automatically controlled processes, from time to time the operator consistently executes several standard procedures to detect possible out-of-bounds situations (procedural recurrent skill). To perform his or her supervisory control task, the operator must be able to use a keyboard (nonprocedural recurrent skill). In the field of computer programming, recurrent skills involve the use of the editor and the interpreter or compiler, the selection of basic language commands, and the application of syntactic rules of the language. In statistical analysis, recurrent skills mainly involve the application of computational algorithms and the generation of applicable sequences of equations.

Nonrecurrent skills, which are sometimes called variable component skills or open skills, are characterized by variable performance over various problem situations. In general, these skills have a nonprocedural character; the exact procedure for executing a particular activity varies each time the task is performed. For instance, after an out-of-bounds situation is detected, its cause has to be diagnosed. It seems clear that the process of diagnosing system failures is highly nonrecurrent, because the failures or combinations of failures, and hence the procedures that lead to a diagnosis typically vary from one problem situation to another. For instance, in computer programming, nonrecurrent skills may involve decomposition of programming problems into subproblems that are increasingly easier to solve and the composition of a program by putting functional parts of programming code together. In statistical analysis, nonrecurrent skills may involve strategic decisions about which statistical techniques, concepts, and principles to use and the coupling of the information that is present in the problem situation to the proper constants and variables.

Skills Analysis

After the two sets of component skills have been identified, an in-depth analysis of all involved component skills has to be conducted. According to the 4C model, different task-analytical techniques should be used for recurrent and nonrecurrent skills.

Component 1: Analysis of recurrent skills. For procedural recurrent skills (e.g., starting up the computer and entering the programming environment, submitting a program to the compiler), a behavioral task analysis (Gropper, 1974) may be used if the skill mainly consists of motor operations; an information processing analysis (Merrill, 1987) may be used if the skill contains relatively many mental operations. Both analyses employ some kind of hierarchical analysis in which a particular procedure is broken down into increasingly more specific steps or actions which are then arranged in the correct order of execution. The results of the analysis may be represented by means of a list of tasks and subtasks (behavioral analysis) or in a flowchart, pseudoprogram, or structured outline (information processing analysis).

For nonprocedural recurrent skills, it is often difficult or even impossible to specify a correct sequence of operations (e.g., handling the keyboard or selecting the correct language commands). Thus, other types of analyses are required. In essence, rules have to be described that specify under which conditions particular actions apply. One possible approach is to conduct a factor-transfer analysis (Reigeluth & Merrill, 1984). In this type of analysis, the factors that vary from situation to situation are identified and coupled to correct actions. The results are represented in so-called decision rules. Another approach is to model performance of the skill in an expert or production system which contains a set of condition-action pairs that specify what should be done under particular circumstances.

For the first component of the presented 4C model, skills are analyzed in terms of "strong methods." The specified procedures and/or decision rules are highly domain specific and basically algorithmic in nature. They have great power because their application warrants that recurrent goals will be obtained. From a psychological viewpoint, recurrent skills are analyzed as if they were automatic processes (Schneider & Shiffrin, 1977; Shiffrin & Schneider, 1977). This is done for a good reason: In the design phase, instructional tactics will be selected that directly foster automation of rules. By automating the recurrent component skills, processing resources become available that may be used to perform nonrecurrent component skills.

Component 2: Analysis of nonrecurrent skills. One way to analyze nonrecurrent skills is to specify the actions and methods that ensure a systematic approach to problem solving (SAP analysis; Mettes, Pilot, & Roossink, 1981). Essentially, a sequence of actions has to be described that characterizes an optimal approach to the kind of problems to be solved. Whereas the results of an SAP analysis can be presented in a flowchart (in this case often called an SAP chart) or structured diagram, it is fundamentally different from an information-processing analysis. An SAP analysis leads to the description of a systematic problem approach that is quite general and has a heuristic nature; an information-processing analysis leads to a description of a task that is highly specific and has an algorithmic nature. For instance, the results of an SAP analysis for decomposition of programming problems may be as follows: Read the problem carefully, mark the input and output conditions, classify the problem, try to recognize the problem as one that is already known, identify the necessary objects, identify the necessary actions to apply to the objects, and so forth (see Tromp, 1989).

Another approach to analyzing nonrecurrent skills is to model the skill in a production system containing domain-independent rules that interpret existing knowledge to solve new problem situations. Thus, whereas recurrent skills are analyzed in terms of strong methods, nonrecurrent skills are analyzed in terms of weak methods. The specified SAPs or rules are less domain-specific and heuristic in nature. Application of these rules does not automatically warrant the obtaining of current goals. Instead, power is exchanged for flexibility. As a result, different problem situations can be solved by the heuristic use of the same available knowledge.

From a psychological viewpoint, nonrecurrent skills are analyzed as if they were controlled processes. There are both theoretical and practical reasons for this choice. From a theoretical point of view, the goal is to develop a description of the exit behavior of the learners. Since the nonrecurrent component skills vary greatly over problem situations, it is often highly implausible that they will be automated by the learners because of the limited time that is typically available for training. From a practical point of view, it is often impossible to foresee all particular problem situations and their related procedures. For instance, thousands of combinations of faults may occur in particular production processes, so that it becomes practically impossible to analyze all related specific rules or procedures: There is always one more bug to describe.

Finally, it should be clear that the efficiency of the application of SAP charts or heuristics depends heavily on the availability of a rich declarative knowledge base. The better organized a learner's knowledge about a particular domain is, the more likely it is that controlled, heuristic processes will lead to a correct solution. In the 4C model, nonrecurrent skills are analyzed mainly in terms of the application of weak methods that make use of a rich declarative knowledge base that is organized in cognitive schemata (see component 4). Consequently, in the design phase, instructional tactics will be selected that foster the acquisition of schemata.

Knowledge Analysis

The goal of the knowledge analysis is to describe declarative knowledge that is either prerequisite or helpful to performance of a complex cognitive skill. According to the 4C model, different forms of knowledge analysis are needed for recurrent and nonrecurrent skills.

Component 3: Analysis of prerequisite knowledge to perform recurrent skills. This component pertains to the analysis of knowledge that is minimally required to be able to perform recurrent skills correctly. Here, the distinction between procedural and nonprocedural recurrent skills is not relevant. Given the kinds of analyses conducted for the first component, the prerequisite knowledge will consist mainly of concepts and facts. Essentially, the knowledge analysis that is to be conducted entails submitting each step in a flowchart, pseudo-program, or structured outline, or each proposition in a decision rule, relative to the question of what the learner has to know in order to be able to perform that step or to apply that rule correctly. In fact, concept analysis (see Tennyson & Cocchiarella, 1986) is also a form of hierarchical analysis because it is repeated for all identified concepts until each of the concepts is one that already has been mastered by the learner prior to the present instruction. Relations between the concepts may be identified in taxonomies (kind-of relations), partonomies (part-of relations), or other hierarchies.

Component 4: Analysis of supportive knowledge to perform nonrecurrent skills. Whereas the knowledge analysis for recurrent skills (component 3) is relatively easy and straightforward, the knowledge analysis for nonrecurrent skills (component 4) is more difficult and elaborated. The knowledge analysis related to nonrecurrent skills results in a description of knowledge that is supportive to performance of this type of skill. It is postulated that often the knowledge will have the form of so-called schemata—cognitive structures that serve as abstractions to assign particular (sub)problems, on the basis of their characteristics, to particular categories that require particular plans or actions for their solution. Thanks to the presence of such schemata, learners are able to interpret unfamiliar situations in terms of their generalized knowledge. Analogy is a process of utmost importance to the use of schemata. This relatively "strong" method may be seen as a mapping process in which the generalized declarative knowledge is mapped on the concrete situation in order to solve a particular problem (Anderson & Thompson, 1987).

For skills that involve *causal reasoning*, schemata that contain causal relations may be described. In a principle-transfer analysis (Reigeluth & Merrill, 1984), the causal relations that underlie the performance of the learner are identified. Complex causal nets can be described in AND/OR graphs (Collins & Stevens, 1983) or other kinds of causal nets. Such graphs enable the presentation of causal dependencies between larger amounts of factors. For instance, in the area of fault management in process industry, AND/OR graphs may describe the causal relations involved in the malfunctioning of a particular controller. The malfunction may be caused by corrosion *or* high temperature; corrosion may be caused by a combination of water *and* oxygen; high temperature may be caused by a failure of a temperature controller, *or* hot weather, *or* a human error, and so forth.

For skills that embody *decision making*, schemata may be described that involve (local) plans to reach particular goals. Essentially, those plans may be viewed as stereotyped, abstract solutions that are related to particular categories of (sub)problems. In a goal-plan analysis (Schank & Abelson, 1977; Soloway, 1985), plans are described and coupled to a hierarchy of goals that must be achieved to solve the problem. For instance, a hierarchy of programming plans can be described in the field of computer programming. High-level programming plans (such as a general input-process-output plan) are related to general goals and may be applied to a very wide range of programming problems; medium-level plans (such as looping structures with an initialization above the loop) are related to less general goals; and low-level plans (such as the blueprint for a statement to print the value of a variable) are related to highly specific goals and applicable to increasingly smaller ranges of (sub)problems.

Finally, many skills involve *qualitative reasoning* (i.e., reasoning involving noncausal terms) about dynamic processes and require that the learner have an abstract notion of the system, machinery, or process that must be manipulated. The term "mental model" is often used to refer to this kind of knowledge (Gentner & Stevens, 1983). Whereas some researchers view this knowledge as fundamentally different from procedural and declarative knowledge, it may also be argued that people have a set of declarative knowledge structures (i.e., schemata) for representing the form, function, and relations between various devices, as well as a set of procedures for reasoning about the interactions among the devices (Anderson, 1988). Whereas the analysis of mental models is in a relatively immature state compared to the analysis of distinct skills and declarative knowledge, it seems important to accurately describe the system, machinery, or process that must be manipulated if nonrecurrent component skills involve qualitative reasoning about dynamic processes.

The Design Phase

After analyzing the complex cognitive skill and the knowledge that is prerequisite or supportive to its performance, an effective training strategy should be composed to be applied in the interactive learning environment that will be used to train the skill. Within any training system for teaching a complex cognitive skill, a distinction can be made between the design of practice (procedural instruction) and the presentation of information (declarative instruction; van Merriënboer & Krammer, 1987). In our view, procedural instruction should be the central part of each training strategy because it relates to the instructional design for actually performing the skill, that is, to the design of the problem(s) or situation(s) which will confront the learner. Then, declarative instruction should scaffold the performance of the skill by the presentation of information that is relevant to its acquisition. Within the 4C model, the components 1′ and 2′ relate to the design of practice and the components 3′ and 4′ relate to the design of information presentation.

The Design of Practice

The effects of practice are twofold. First, as a result of practice, compilative processes (e.g., Anderson, 1983, 1987) produce domain-specific procedures that eventually may be applied directly to performance of the skill without the intercession of weak heuristic methods. This process of rule automation is particularly important to the first component of our model: the training of recurrent skills. Second, as a result of practice, inductive processes (e.g., Carbonell, 1984, 1986) may create new schemata or adjust existing schemata to make them more in tune with experience. This process of schema acquisition is particularly important to the second component of our model: the training of nonrecurrent skills.

Component 1′: Selecting tactics that promote compilative processing. Component 1′ pertains to the design for practicing recurrent skills; the instruction is designed to promote a rapid development of domain-specific, automated rules. Anderson (1983, 1987) identified knowledge compilation as an important process in the creation of procedures that eventually may directly control behavior. It includes both the incorporation of newly acquired knowledge in task-specific procedures and the "chunking" of procedures that consistently follow one another during performance of the skill. Knowledge compilation speeds performance considerably and implies a reduction of processing load because newly acquired knowledge need no longer be retrieved from memory and held active to be interpreted for more general procedures. With regard to the design of practice, it should be noted that knowledge compilation is an elementary cognitive process that is not subject to strategic control; it is primarily a function of the amount of consistent practice.

Instructional tactics for the design of training that promote rapid compilation usually invite learners to repeat performance mechanically and consistently. These tactics often are associated with repeated imitation and drill. If the algorithmic procedures or specific rules that specify a correct performance are known (component 1), the most common method to be used is single-step or step-by-step instruction. For instance, one may teach multiplication by providing each step of the multiplication procedure to the learner and having the learner imitate each step until the procedure is completed. By repeating this process over and over, the need for presentation of the steps disappears and the procedure ultimately becomes automatic.

For recurrent skills that are difficult and demanding, part-task and part-whole training techniques form the most common solution to practicing these skills. Under this approach, the whole procedure is decomposed into parts and learners are trained extensively on each part separately before they move to practicing the whole procedure. The procedure may be broken down by segmentation (distinct temporal or spatial parts), fractionation (distinct functional parts), or simplification (simpler to more complex parts; Wightman & Lintern, 1985). For instance, Landa (1983) breaks down complex procedures by segmentation and suggests practicing the parts in a forward-chaining manner (the "snowball approach"); Scandura (1983) breaks down the procedure by simplification (simple to complex paths in the whole procedure) and suggests practicing the parts from simple to complex. To conclude, whereas many part-task approaches exist, they all aim at a smooth automation of the complete skill by practicing its parts in a prespecified sequence.

Component 2′: Selecting tactics that promote inductive processing. Practice not only results in the development of domain-specific procedures, but also in the modification of declarative knowledge (Proctor & Reeve, 1988). Component 2′ relates to the design for practicing nonrecurrent skills; the instruction is designed to promote the (re)construction or (re) organization of underlying knowledge in schemata that subsequently may be used by analogy to solve novel problem situations. Schema induction seems an important process in the adjustment of existing schemata to make them more in tune with experience, or in the construction of new knowledge in the form of schemata. For example, a more generalized schema may be produced if a set of successful solutions is available for a class of related problems, then a schema may be created that abstracts from the details. A more specific schema may be produced if a set of failed solutions is available for a class of related problems, then particular conditions may be added to the schema which restrict its range of use.

After useful schemata have been developed, they may be used by analogy to generate behavior in new, unfamiliar problem situations. Obviously, this will often be the case if no, or insufficient, task-specific, automated procedures are available. With regard to the design of practice, it should be noted that inductive processing seems to be a process that is—at least to a certain degree—subject to strategic control; thus, it may require the conscious attention and mindful abstraction from the learner. In addition, it should be noted that, like all declarative knowledge, schemata that are formed by inductive processes may be compiled into domain-specific procedures. Obviously, this will be the case only if a schema is consistently used over problem situations. Then, the compiled procedures eventually may directly produce the effect of the use of the schema by analogy (or other weak methods) without making reference to declarative knowledge (Lewis & Anderson, 1985).

From the literature, many instructional tactics are known that may have a positive effect on inductive processing, and thus on the acquisition of schemata during practice. Whereas the following list is far from complete, it gives a good overview of some important instructional tactics that can be used for the design of practice in such a way that it promotes inductive processing.

VARIABILITY OF PRACTICE. The application of variability of practice is one of the best-known instructional principles to encourage learners to develop schemata, and one that almost consistently has resulted in beneficial effects on transfer of training (for examples, see Cormier & Hagman, 1987; Singley & Anderson, 1989). According to this tactic, nonrecurrent component skills must be practiced under conditions that require the performance of different variants of the task over problem situations.

CONTEXTUAL INTERFERENCE. Closely related to the first tactic is the application of contextual interference (Lee & Magill, 1985; Shea & Zimny, 1983). Application of this tactic implies that nonrecurrent component skills be practiced under conditions of high interference. This may be realized by repeatedly presenting a restricted set of problems that each require different solutions in a random order. Variability of practice generally is considered the determinant factor in transfer of training; however, the results of studies on contextual interference suggest that it is not only the variability per se that determines the extent of transfer of training, but also the way it is structured across acquisition problems.

RULE-BASED INSTRUCTION. Although this term may be misleading, it is used by Fisk and Gallini (1989) to indicate the presentation of different types of problems with common structural features, together with integrated schemes of procedural and factual information. Thus, single components of instruction consist of information that is organized in schemes and different types of problems which can be solved by the use of the presented schemes. In the study by Fisk and Gallini, rule-based instruction led to higher transfer of acquired skills than single-step instruction.

EMPHASIS MANIPULATION APPROACH. Gopher, Weil, and Siegel (1989) studied an innovative practice strategy that manipulates the emphasis or priority of selected subcomponents of a task, or, in terms of the 4C model, aspects of the nonrecurrent skill. During practice, subjects are directed to pay full attention to important aspects of the skill one at the time, and to pay attention to all aspects of the skill only after quite extensive practice. According to Gopher et al., the approach is suitable to teach subjects strategies and lead them to incorporate those strategies in long-term schemata.

HIERARCHICAL APPROACH. This approach to practice (Frederiksen & White, 1989) is based on theories of mental models. According to the approach, a set of "problem environments" must be devised to focus on particular component skills, concepts, and strategies that are important to the whole skill. Practice in the problem environments takes place in an order that follows from the hierarchical relations among these components and aims at the development of the trainee's mental model.

PROBLEMS WITH NON-SPECIFIC GOALS. In the field of mathematics, Owen and Sweller (1985) demonstrated that preventing the use of means-ends analysis by reducing goal specificity of presented problems resulted in higher schema acquisition and improved transfer performance. In kinematics and geometry, Sweller (1988; Sweller, Mawer, & Ward, 1983) found evidence that problems that do not include specific goals facilitate inductive processing more than conventional problems and lead to higher transfer of acquired skills.

COMPLETION STRATEGY. Van Merriënboer (1990a, 1990b) proposed a form of practice in which the basis of training is not the solving of increasingly complex problems but the completion of increasingly larger parts of incomplete solutions. By presenting this kind of problems, students are forced to study the (incomplete) solutions carefully in order to solve the problem. This is believed to promote inductive processing and schema acquisition because students abstract from the details of the incomplete solutions presented.

Annotated, worked examples. Another powerful procedural tactic that may promote schema acquisition is the presentation of problems in direct combination with worked examples for similar problems; the worked examples provide a blueprint to approach the problems to be solved. Anderson, Boyle, Corbett, and Lewis (1986) suggest that schema induction is further facilitated when the critical features in these examples are clearly identified, that is, when the examples are annotated with information about what they are supposed to illustrate.

The Design of Information Presentation

Two aspects can be distinguished with respect to the presentation of information. First, information has to be presented that is directly conditional to a correct performance of recurrent skills. It may be presented in such a way that it is restrictedly encoded. In the analysis phase, this information has been identified as the components 1 and 3. Second, information has to be presented that is helpful to a correct performance of nonrecurrent component skills. It should be presented in such a way that it is elaboratively encoded, that is, connected to already existing knowledge so that it yields schematic knowledge that may be used by analogy or other controlled processes to solve new problem situations. In the analysis phase, this information has been identified as components 2 and 4.

Component 3': Selecting tactics that promote restricted encoding. For the design phase, this component pertains to the presentation of information that is relevant to the acquisition and performance of recurrent skills. As for component 1', the instruction is designed to promote the rapid development of domain-specific, automated rules. For compilative processing to occur, the necessary information must be active in working memory while the problem is being solved. This can be achieved by making the information directly available during problem solving or by retrieving the knowledge from long-term declarative memory. In the second case, the knowledge may be declaratively encoded in a relatively isolated manner; it is not critical that it be embedded into existing schemata so that during acquisition no particular reference has to be made to related knowledge.

The question of what information to present to the learners has mainly been answered during the analysis of components 1 and 3. First, it relates to the procedures or specific rules that describe the correct performance of the recurrent component skill (component 1); second, it relates to the facts or concepts that form an integral part of these strong methods (component 3). Usually, a "didactical specification" is needed to transform this information into a form that is suitable to present to the students (Resnick, 1976). Essentially, this didactical specification pertains to the formulation of rules, procedural steps, concepts, or facts in terms that are clearly understandable to the learners for whom the instruction is designed.

Instructional tactics for the presentation of information that promotes restricted encoding may aim either at the prior availability of the information in declarative memory or its direct availability to the learners during problem solving. If prior availability is desirable, the learner can be encouraged to maintain the new information in an active state just by repeating it over and over, either aloud or mentally, in order to store it in declarative memory. This mere repetition of information, or rehearsal, is sometimes associated with rote learning. However, in difficult tasks for which excessive working memory load is expected, retrieval of the information from declarative memory may become difficult. In such cases, it is advisable to explicitly present the new information only when it is needed; that is, parts of the information are provided precisely at the moment they are needed during practice.

According to the 4C model, three key aspects to the presentation of information are *partitioning*, *demonstration*, and *scaffolding*. Partitioning is particularly important for recurrent skills because only the presentation of relatively small amounts of new information at the same time can prevent processing overload. In addition, this information should have to be held in working memory a relatively short time in order to prevent working memory failures that may result in incorrect compilations and negatively affect performance of the skill. Thus, the necessary declarative instruction should be compacted into its bare essentials and gradually elaborated during the skill-acquisition process.

This principle pertains to the presentation of the specific rules or procedures ("how-to" instruction) as well as to the presentation of necessary concepts and facts. For instance, a support or help system may be used in a step-by-step paradigm, as discussed in the previous section. During practice, the system provides each applicable procedural step or rule at the moment it must be applied; in addition, the facts and concepts that are used in that step or rule are simultaneously explained.

If it is not possible to provide the necessary partitions of information directly during practice, one should strive to present them in such a way that they are easily accessible during practice. In agreement with this point of view, Carroll, Smith-Kerker, Ford, and Mazur-Rimetz (1986, 1988) showed that standard text-editor manuals can be made much more effective if they have the form of an easily accessible "minimal manual" that is shortened and focused just on the information that is necessary to perform the skill.

Another tactic for the presentation of information that is relevant to the performance of recurrent skills relates to demonstration. In general, one should provide not only the rules, procedures, or concepts that are relevant to performing the skill (so-called "expository generalities"; Merrill, 1983), but also demonstrations that illustrate the application of the rules and procedures as well as examples that typify concepts ("expository instances"). Apparently, no generalities exist for the plain facts that have to be presented.

Finally, the last principle pertains to scaffolding, which means that the presentation of information becomes increasingly superfluous to the performance of recurrent component skills as the learners gain expertise (e.g., Charney & Reder, 1986). For instance, in a support or help system, one may first systematically present supportive information during practice, then provide that information only on request of the learner, and finally provide no supportive information at all.

Component 4′: Selecting tactics that promote elaborative encoding. This component relates to the presentation of information that is relevant to the acquisition and performance of nonrecurrent component skills. As for component 2′, the instruction is designed to promote the acquisition of schemata that subsequently may be used to solve new problem situations. For elaborative encoding to occur (i.e., enrichment of newly presented information by existing knowledge), the information must be integrated and linked to existing knowledge structures in order to provide a knowledge base that is highly organized and suitable to be operated upon by weak methods. Elaborative encoding processes connect schemata with the new information and they infer from the schemata information that was not presented in the instruction. The result is an increase of interconnections in and between schemata, which is thought to facilitate retrievability and useability because multiple retrieval routes to particular information become available. What learners already know about a topic is used to help them structure and understand new information about that topic, so that learners are encouraged not only to rehearse the new information, but in addition are stimulated to relate it to previous knowledge and form either new or adjusted schemata.

For the most part, the question of what information to present to the learners has been answered during the analysis of components 2 and 4. First, it relates to the heuristics (domain-general rules, rules of thumb) and SAP charts that describe effective problem approaches for nonrecurrent skills (component 2). Second, it relates to the causal nets, local plans, and other schematic structures, such as process or machine models, that underlie a productive application of these weak methods (component 4).

With regard to the teaching of SAP charts or heuristics, most researchers agree that it is helpful to present this information to the learners after a didactical specification (e.g., Mettes et al., 1981; Schoenfeld, 1979). However, with regard to the underlying schemata, there is a vivid discussion among researchers who favor particular forms of either expository learning or (guided) discovery learning (McDaniel & Schlager, 1990). For instance, in the field of computer programming, one might aim at the acquisition of schemata by explicitly presenting programming plans to students (e.g., Soloway, 1985) or by providing the students many worked examples, in the form of concrete programs, during practice to facilitate the acquisition of these schemata by inductive processing.

According to the 4C model, the discussion on discovery learning and expository learning is obscured by the fact that the components 2′ and 4′ often are not distinguished. On the one hand, instructional strategies that favor problem-solving activities in some form of (guided) discovery learning might well lead to useful schemata, provided tactics are applied that promote inductive processing during problem solving (component 2′). This point of view is present in the fields of "situated cognition" and "constructivism," which claim that learners construct meaning (i.e., acquire schemata) from interacting with real events or with interactive environments that simulate real events. Indeed, this approach warrants that new information about a particular domain be fully integrated with already existing knowledge. On the other hand, there seems to be no strong reason whatsoever for *not* simultaneously providing the learner with prespecified causal nets, local plans, or models that may be helpful and offer guidance to solve the problems in that particular domain (component 4′). The point is that these approaches need not be seen as distinct alternatives, but merely as two aspects of instruction that can, and often should, complement each other.

In contrast to tactics that promote restricted encoding, instructional tactics that promote elaborative encoding of presented schema-like information focus primarily on the prior availability of this information in declarative memory in such a way that it is highly integrated with knowledge that already exists. This increases the chance that the information can be helpful to the performance of nonrecurrent skills. Together with the intrinsic complexity of the information, this leads to implications for partitioning the information. The amount of information that must be presented at the same time can be considerable. For instance, SAP charts, causal nets, or process models (e.g., a concrete computer model, a model of a distillation process) can be quite complex. The information usually will be presented before units of practice and instructional tactics for information presentation have to be used that promote elaborative encoding of the presented material. Such tactics encourage the learner to relate the new information to existing knowledge; that is, they promote meaningful learning by facilitating the assimilation of the new information to schemata that already exist in memory. A wide variety of such tactics has been put forward (e.g., Annett & Sparrow, 1985; Brooks & Dansereau, 1987; Mayer & Greeno, 1972; Merrill, 1983; Reigeluth, 1983b). For instance, well-known tactics are the application of advance organizers, metaphors, and mnemonic systems and the encouragement to paraphrase particular pieces of information.

With regard to demonstration and scaffolding, essentially the same principles apply as for recurrent skills. Thus, one should not only present heuristics, SAP charts, or schema-like units that are relevant to performing the skill, but also provide demonstrations that illustrate the application of the heuristics or SAP charts, as well as concrete examples that characterize, for instance, causal nets (particular predictions that can be made), plans (particular instantiations of plans), or models (particular states of a process). Given the high level of abstraction as compared to the information that is relevant to the performance of recurrent skills, it may be difficult to develop adequate demonstrations. Modelling (either on the job or in an instructional setting) through close observation of an expert or instructor who is performing the nonrecurrent skill and explaining why he or she is doing each step is often an effective approach to the demonstration of nonrecurrent skills.

Finally, it should be noted that systematically decreasing the amount of scaffolding is less easy than for recurrent skills. This is due to the fact that the presented information *may* help solve the current problem but does not necessarily do so. In most cases, it seems advisable to present the information that seems helpful to perform particular skills before units of practice and to keep this information available for the learners during the whole training process.

Composing the Training Strategy

Whereas the composition of the training strategy is, to a certain degree, a creative process, the selected instructional tactics will heavily constrain the problem space of instructional design. First, the selected tactics for the design of practice guide the design of the interactive learning environment that is developed to practice the skill. Particular tactics prescribe methods that must be applied to secure compilative processing for recurrent component skills, while other tactics prescribe methods to secure inductive processing during the performance of nonrecurrent skills.

Second, the selected tactics for the presentation of information give guidance to designing the amount and form of the presented information as well as to the timing of its presentation within the learning environment. Particular tactics prescribe methods to promote restricted encoding of information that is prerequisite to a correct performance of recurrent skills, while other tactics prescribe methods to promote elaborative encoding of information that is presented to support the performance of nonrecurrent skills.

Reflective Expertise and Transfer: A Basic Prediction

According to the 4C model, reflective expertise is best characterized as the ability to solve new problems on the basis of two highly interrelated cognitive processes. First, domain-specific procedures are available to solve familiar aspects of a problem. Second, heuristics may be applied to interpret the current situation in terms of a rich declarative knowledge base, that is organized in schemata, to solve unfamiliar aspects of the problem. Thus, the first element is related to components 1 and 3 (rule automation) and the second element is related to components 2 and 4 (schema acquisition).

The processes are interrelated in two ways. On the one hand, the application of domain-specific procedures frees up processing resources so that heuristics can be applied. On the other hand, the rich declarative knowledge base enables learners to *reflect* on the solution that has been reached by the application of domain-specific procedures. According to this line of reasoning, reflective expertise in performing a complex cognitive skill is clearly more than the sum of its automatic parts (see also Olsen & Rasmussen, 1989). If unskilled performance were simply less automatic than skilled performance, unskilled performers would be able to do whatever skilled performers could do, only less automatically, and clearly this is wrong.

In presenting their ADAPT design model, a forerunner of the 4C model, Jelsma, van Merriënboer, and Bijlstra's (1990) extensively discussed the implications for transfer of training. In short, two processes seem to be relevant to transfer of training: procedural overlap and knowledge-based transfer. With regard to procedural overlap, it is assumed that transfer from a learned task to a transfer task will be positive to the extent that the underlying procedure sets overlap. Obviously, this component is particularly important to recurrent skills within the whole complex cognitive skill. The identification of the common set of specific rules or procedures allows quantitative predictions for transfer of training (e.g., Polson & Kieras, 1984; Singley & Anderson, 1988). Furthermore, the concept of procedural overlap naturally implies that there is also a set of procedures that is specific to the learned tasks and the transfer tasks.

If the size of the specific set is large—that is, if the transfer task involves numerous deviations from the original training task—transfer of training can no longer be attributed exclusively to procedure overlap. Then, knowledge-based transfer may explain transfer to the extent that declarative knowledge structures, or schemata, are available from other problem-solving situations that may help to solve the nonfamiliar aspects of the current problem situation. Thus, for this component, schemata provide a basis for transfer between different uses of the same knowledge. Obviously, this component is particularly important to the nonrecurrent skills that form part of the whole complex cognitive skill.

In the 4C model, analogy is seen as a particularly important process to provide knowledge-based transfer; it is conceptualized as the mapping process in which schemata are used to solve new problems. In fact, analogy draws comparisons between the current problem situation and possibly relevant, generalized knowledge from other more or less similar problem situations. For example, in the field of computer programming, one may solve the problem of how to compute an average on the basis of analogy to a general schema for iteration that is developed while solving other problems involving looping structures. If no relevant knowledge is available, analogy fails and other less powerful processes must be used to perform the transfer task. In the worst case, it may happen that no solution can be found.

In agreement with the ADAPT design model, the 4C model postulates that the importance of knowledge-based transfer increases as the transfer becomes farther away from the training task, that is, as the size of the overlapping set of procedures decreases and the specific set increases. Furthermore, it is assumed that the processes that are responsible for knowledge-based transfer (and in particular, analogy) will be more successful to the extent that the knowledge is better organized into schemata. This declarative knowledge representation in schemata is considered to be a sound basis for analogy to operate upon because multiple retrieval routes to particular information are available which offer the possibility to derive relevant information more easily. In other words, the relative importance of schema acquisition increases if the transfer tasks become more different from the original training tasks.

This leads us to the basic predictions of the 4C model. In contrast to most other models, the 4C model explicitly takes the knowledge-based component for performing complex cognitive skills into account by identifying SAP charts and heuristics as well as schemata that may help to make an effective use of them (components 2 and 4). In addition, instructional tactics are selected that promote adequate schema acquisition by inductive processing and elaborative encoding (components 2' and 4'). Given these facts, it is hypothesized that the model opens the way to attain reflective expertise and hence reach good transfer performance. Specifically, it is predicted that the superiority of instruction that is developed according to the 4C model will become more apparent for performing transfer tasks that are more different from the training tasks. Illustrations of applications of the model that give support to this basic prediction are presented in the next section.

APPLICATIONS OF THE FOUR-COMPONENT MODEL

Up to the present, the ideas from the 4C model have been applied to design instruction in the fields of fault management in process industry, introductory computer programming, and statistical analysis. In all three domains, experiments have been conducted to test the model's predictions regarding reflective expertise and transfer, namely, that the application of the model will yield instruction that leads to higher transfer performance than conventional instruction, and that this superiority will increase as the transfer tasks differ more from the original training tasks. The results of the conducted experiments are briefly discussed in the following sections.

Fault Management

The training of process operators for fault management tasks is a complicated matter. In fault management training, operators must learn to detect, diagnose, and compensate for a countless number of combinations of system failures that can occur in the process to be controlled. It is implausible to cope with all possible system failures or combinations of failures in one training program, and, even if this could be realized, it would be impossible to exactly anticipate all kinds of system failures. Yet it is precisely for the purpose of handling these unforeseen situations that human operators are employed. Clearly, for skilled operators, reflective expertise is a prerequisite to adequate performance of their complicated task.

Within the framework of the 4C model, training strategies were designed which are best characterized as "procedure-based" training (see, Morris & Rouse, 1985; Shepherd, 1986; Shepherd, Marshall, Turner, & Duncan, 1977). The training simulator PROCESS (Program for Research on Operator Control in an Experimental Simulated Setting;* Jelsma & Bijlstra, 1990) was used to conduct experiments on these training strategies (Jelsma, 1989; Jelsma & Bijlstra, 1988). In one of the experiments, the selected faults were introduced in the simulation program and their related procedures were practiced extensively during several days. Using PROCESS, it was possible to predefine both the types of faults and the time at which they should occur in the simulated water-alcohol distillation process. The experimental and control training strategies differed with respect to the level of contextual interference under which the handling of the selected faults was practiced (component 2′, inductive processing). Low contextual interference involved the training of particular categories of faults in a blocked practice schedule (this tactic is *not* recommended by the 4C model) and high contextual interference involved the training of these faults in a random order (recommended by the 4C model). The two strategies were identical with regard to the other three components of the 4C model.

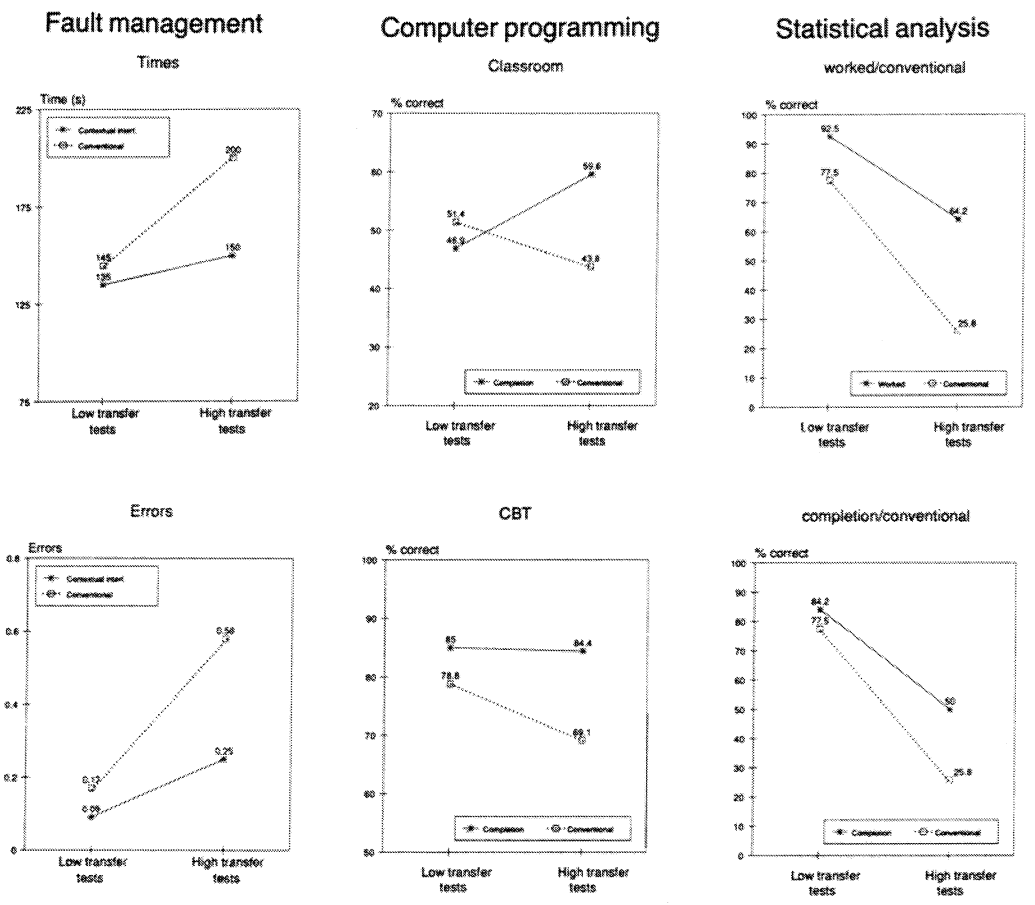
As shown in the left column of Figure 2, training under high contextual interference conditions proved to lead to higher transfer performance (i.e., faster times and higher accuracy) than training under low contextual interference conditions. In addition, when the procedural overlap between the learned tasks and the transfer tasks decreased, the superiority of the high contextual interference condition over the low contextual interference condition increased. This finding is fully in agreement with the predictions of the 4C model and supports the view that practicing under high contextual interference conditions facilitates schema acquisition by inductive processing. In general, the selection of instructional tactics as suggested in the 4C model proved to be useful to the design of an adequate training strategy for operator training. In fact, the data show that the application of the model contributed to the development of reflective expertise, because the learners were better able to manage faults not encountered previously.

Computer Programming

In most conventional introductory programming courses, practice mainly consists of the

*A PC-version of PROCESS is available from Bijlstra & van Merriënboer, Training Consultancy and Development, P.O. Box 217, 7500 AE Enschede, The Netherlands.

FIGURE 2 □ Scores on Low- and High-Transfer Tests. Scores are shown for: fault management instruction under high and low contextual interference (time and error scores), programming instruction under the completion strategy and a conventional strategy (classroom study and CBT), and statistical analysis instruction under the worked-examples strategy, the completion strategy, and a conventional strategy (worked vs. conventional and completion vs. conventional).



generation of increasingly complex computer programs. In these courses, learning outcomes and transfer performance are usually low (for an overview, see Linn, 1985; Pea & Kurland, 1984). Within the framework of the 4C model, an alternative form of training was designed which has been referred to as the "completion strategy" (van Merriënboer & Krammer, 1990; van Merriënboer & Paas, 1989). In this strategy, the students start with the running and hand-tracing of existing computer programs, then complete increasingly larger parts of well-structured, easily readable, but incomplete computer programs, and finally generate programs on their own. Thus, instead of the generation of increasingly complex computer programs, the completion of increasingly larger parts of incomplete programs forms the kernel of the training. The completion strategy provides examples in the form of incomplete computer programs that are directly available during practice and must be studied carefully in order to be able to correctly finish them; thus, they are expected to facilitate schema acquisition by requiring inductive processing during problem solving.

In two experiments, the effectiveness of the completion strategy was compared to the effectiveness of conventional problem-oriented strategies. In both a classroom study (van Merriënboer, 1990a, 1990b) and a subsequent study on computer-based training (CBT; van

Merriënboer & De Croock, in press), the control group received worked examples that illustrated new information on features of the programming language and their syntactical details, and conventional programming problems that required the design and coding of complete computer programs. The experimental group received completion assignments that consisted of an incomplete program together with new information on programming language features that were illustrated by the incomplete program, as well as instructions to complete, extend, or change the program to meet a given problem specification.

As may be seen in the middle column of Figure 2, the completion strategy yielded transfer performance higher than or equal to conventional training strategies in both the classroom study and the CBT study. Low-transfer tests measured the knowledge of the commands, syntax, and standard language constructs of the programming language; high-transfer tests measured proficiency in the design and construction of programs for *new* programming problems. For the high-transfer tests, the completion strategy was significantly superior to the conventional strategies in both studies. Thus, as predicted by the 4C model, the superiority of the completion strategy over conventional strategies increased as the tests required more transfer of acquired skills. In further analyses, it was found that the students who worked according to the completion strategy showed a superior use of programming language templates. As learned templates are a special kind of schemata, this indicates that the completion strategy indeed facilitated schema acquisition and so improved transfer performance.

Statistical Analysis

Statistical analysis involves more than the straightforward application of computational algorithms. The problem situations are infinitely variable and each situation poses its own requirements as to the concepts and techniques that are optimal to use. For instance, learners may well be able to compute the mean, mode, and median on a fixed array of numbers. However, statistical analysis also means that they must be able to decide on the statistics that are optimal to use and to compute these statistics in complex situations in which the numbers are not prespecified in a fixed array but must be searched for in complex sets of data. Thus, training for reflective expertise is important to enable learners to make their statistical knowledge transferable. However, substantial evidence is available that conventional training strategies that focus on solving goal-specific open problems are not effective to reach such transfer (Cooper & Sweller, 1987; Larkin, McDermott, Simon, & Simon, 1980; Sweller, 1988, 1989; Sweller, Chandler, Tierney, & Cooper, 1990; Tarmizi & Sweller, 1988).

Within the framework of the 4C model, Paas and van Merriënboer (1992) developed two alternative training strategies for statistical analysis. One strategy was based on the use of worked examples and the other on the use of completion assignments during practice. Both tactics are recommended for component 2′ of the 4C model and thus are assumed to promote inductive processing. The two strategies were compared to a conventional strategy. All strategies were implemented in a CBT program that was designed to teach statistical analysis skills to secondary school students. All students were offered identical formal instruction and an identical set of statistical problems. For the conventional training strategy, all problems in the set were goal-specific, open problems that students had to solve on their own. For the worked strategy, the first two problems of each subset of three consisted of problems together with worked-out problem approaches and solutions, while the third problem in each subset was an open problem. For the completion strategy, the first two problems in each subset of three were problems together with incomplete solutions that had to be completed, while the third problem was, again, an open problem.

As shown in the right column of Figure 2, both the strategy that was based on worked examples (upper part) and the strategy based on completion assignments (lower part) yielded higher transfer performance than the conventional strategy. Low-transfer tests measured the proficiency to solve statistical problems that were highly similar to the problems used during training; high-transfer tests pertained to problems that significantly differed from those used during training. For both tests, training strategies based on worked examples or completion assignments were superior to the conventional strategies, and this superiority increased if the tests required more transfer of acquired skills. These data support the predictions of the 4C model and are in agreement with the results obtained in the fields of fault management and computer programming.

To conclude this section, elements of the 4C model proved to be very useful to the conduct of instructional design for teaching complex cognitive skills in three different domains. Whereas the model is still formulated on a general level, its fundamental concepts suggest innovative training strategies that promote the development of reflective expertise and the acquisition of transferable skills.

DISCUSSION

A four-component instructional design model for the training of complex cognitive skills is presented in this article. The complex skill is decomposed into a set of recurrent skills that remain highly consistent over various problem situations and a set of nonrecurrent skills that require variable performance over different situations. In the analysis phase, the four components relate to (1) skills analyses to describe the recurrent skills in terms of "strong methods," (2) skills analyses to describe the nonrecurrent skills in terms of "weak methods," (3) knowledge analyses to describe the knowledge that is prerequisite to performance of recurrent skills, and (4) knowledge analyses to describe the knowledge that may be helpful to performance of nonrecurrent skills.

In the design phase, instructional tactics are selected for each of the four components. Here, the components relate to (1′) the design of practice to promote knowledge compilation for recurrent skills, (2′) the design of practice to promote inductive processing during the performance of nonrecurrent skills, (3′) the design of the presentation of information that is prerequisite to correctly perform recurrent skills, and (4′) the design of the presentation of information that may be helpful to performance of nonrecurrent skills.

Finally, the composition of the training strategy is governed by the selected instructional tactics; they guide the design of the interactive learning environment and the information that is presented herein.

The basic prediction of the 4C model is that its application leads to "reflective expertise" and, consequently, to increased transfer performance. The "expert" aspect pertains to the ability to solve familiar aspects of a problem by highly domain-specific procedures. In addition, these procedures free up processing resources that are necessary for reflection to occur. The "reflective" aspect pertains to the ability to solve unfamiliar aspects of a problem by heuristic processes that make use of a rich declarative knowledge base which is organized in schemata. This rich knowledge base also yields a sound basis for reflection on the quality of found solutions. Thus, the first aspect of reflective expertise is related to the components 1 and 3 (rule automation) and the second aspect to the components 2 and 4 (schema acquisition).

Applications that fit the model in the fields of fault management in process industry, introductory computer programming, and statistical analysis gave some support to the basic prediction. The instruction developed in accordance with the 4C model yielded higher transfer performance than control conditions. This superiority became more evident as the transfer tasks differed more from the training tasks. However, these studies clearly did not test the complete model, but only particular elements of it. Specifically, they focused on the second component, the application of particular instructional tactics that promote schema acquisition by inductive processing.

Clearly, more research in instructional science is needed to test other elements of the 4C model. Such research should aim at both the analysis and the design phase of the model. For the analysis phase, an important research question pertains to the decomposition of the skill in recurrent and nonrecurrent component skills. For some complex cognitive skills, it is the authors' experience that it may prove to be very difficult to conduct an optimal decomposition of the skill because there is a lack of good models for principled skill decomposition.

Another research question is related to the analysis of complex knowledge structures, and, in particular, the knowledge that underlies the ability to mentally simulate and qualitatively reason about dynamic processes. Whereas such qualitative process models seem particularly important to the performance of complex cognitive skills that encompass trouble shooting or fault management, the methods for analyzing such models are in a relatively immature state compared to the methods for analyzing skills, causal nets, and plans.

In the design phase, a first concern for future research should be the extension, formulation, and confirmation of instructional tactics that possibly can be applied in each of the four components. Tactics can be formulated in a goals-circumstances-method format (van Merriënboer & Krammer, 1987). The goals are related to the desired outcomes of the instruction, the circumstances are factors that influence the effects of methods but cannot be manipulated, and the methods are manipulations to achieve different outcomes under different circumstances. Each instructional tactic should have at least one goal, often one or more circumstances to delimit its validity, and exactly one method.

In the present article, the goals are only loosely described in terms of the cognitive processes that are promoted by classes of instructional tactics that are related to one of the four components: compilative processing, inductive processing, restricted encoding, and elaborative encoding. In addition, the circumstances under which those tactics should be applied often are not exactly described. Obviously, research should lead to a refinement of the goals and the circumstances in tactics to make an unambiguous selection of tactics for each of the four components possible. As a related concern, descriptive models must be developed to guide the combination and integration of selected instructional tactics in the composition of training strategies that may be applied in interactive learning environments.

One may ask what significance the presented framework has for cognitive science. Given the 4C model and the results that have been obtained by its application, the learning of heuristics and schematic, generalized declarative knowledge that may help to perform the nonrecurrent components of complex cognitive skills (components 2 and 4) seems to be essential for the acquisition of such skills. This has direct implications for cognitive theories, and in particular for current theories of skill acquisition that describe the cognitive processes involved in learning complex cognitive skills. In particular, these theories focus on learning by doing (i.e., learning conventional problem-solving strategies) and rule automation (components 1 and 3) as the most essential processes in the acquisition of such skills. They often seem to be incomplete with regard to their description of the acquisition of highly organized declarative knowledge structures or schemata.

To sum up, it seems that instructional design may contribute to cognitive science, because the study of instructional design models such as the 4C model and their resulting training strategies may provide evidence for the importance of particular cognitive processes that are neglected or underestimated by theories in cognitive science. Eventually this may lead to the revision or extension of those cognitive theories. Whereas instructional design cannot lead to the "discovery" of new cognitive processes or a highly detailed description of those processes, the claim stands that studies in instructional design may provide evidence for the importance of particular cognitive processes, and thus lead to the integration of different perspectives in cognitive science. Such an integration of cognitive viewpoints is badly needed in order to be able to explain and predict human cognition and performance in newly developed instructional systems. ☐

Jeroen J. G. van Merriënboer and Fred G. W. C. Paas are with the University of Twente, Department of Instructional Technology, and Otto Jelsma is with the Logistic Training Division of Fokker Aircraft BV, the Netherlands.

REFERENCES

Anderson, J. R. (1983). *The architecture of cognition*. Cambridge, MA: Harvard University Press.

Anderson, J. R. (1987). Skill acquisition: Compilation of weak-method problem solutions. *Psychological Review, 94*, 192–210.

Anderson, J. R. (1988). The expert module. In M. C. Polson & J. J. Richardson (Eds.). *Foundations of intelligent tutoring systems* (pp. 21–53). Hillsdale, NJ: Lawrence Erlbaum.

Anderson, J. R., Boyle, C. F., Corbett, A., & Lewis, M. (1986). *Cognitive modelling and intelligent tutoring* (Tech. Rep. No. ONR-86/1). Pittsburgh, PA: Carnegie-Mellon University, Dept. of Psychology.

Anderson, J. R., & Thompson, R. (1987). *Use of analogy in a production system architecture* (Tech. Rep.). Pittsburgh, PA: Carnegie Mellon University, Dept. of Psychology.

Annett, J., & Sparrow, J. (1985). Transfer of training: A review of research and practical implications. *Programmed Learning and Educational Technology, 22*, 116–124.

Brooks, L. W., & Dansereau, D. F. (1987). Transfer of information: An instructional perspective. In S. M. Cormier & J. D. Hagman (Eds.). *Transfer of learning: Contemporary research and applications* (pp. 121–150). San Diego, CA: Academic Press.

Carbonell, J. G. (1984). Learning by analogy: Formulating and generalizing plans from past experience. In R. S. Michaelsky, J. G. Carbonell, & T. M. Mitchell (Eds.), *Machine learning: An artificial intelligence approach* (Vol. 1, pp. 137–161). Berlin: Springer-Verlag.

Carbonell, J. G. (1986). Deprivational analogy: A theory of reconstructive problem solving and expertise acquisition. In R. S. Michaelsky, J. G. Carbonell, & T. M. Mitchell (Eds.), *Machine learning: An artificial intelligence approach* (Vol. 2, pp. 371–392). Los Altos, CA: Morgan Kaufman.

Carroll, J. M., Smith-Kerker, P. L., Ford, J. R., & Mazur-Rimetz, S. A. (1986). *The minimal manual* (IBM Research Rep. 11637). Yorktown Heights, NY: IBM Watson Research Center.

Carroll, J. M., Smith-Kerker, P. L., Ford, J. R., & Mazur-Rimetz, S. A. (1988). The minimal manual. *Human-Computer Interaction, 3*, 123–153.

Charney, D. H.., & Reder, L. M. (1986). *Initial skill learning: An analysis of how elaborations facilitate the three components* (Tech. Rep.). Pittsburgh, PA: Carnegie-Mellon University, Dept. of Psychology.

Collins, A., & Stevens, A. L. (1983). A cognitive theory of inquiry teaching. In C. M. Reigeluth (Ed.), *Instructional design theories and models* (pp. 247–278). Hillsdale, NJ: Lawrence Erlbaum.

Cooper, G., & Sweller, J. (1987). Effects of schema acquisition and rule automation on mathematical problem-solving transfer. *Journal of Educational Psychology, 79*, 347–362.

Cormier, S. M., & Hagman, J. D. (Eds.). (1987). *Transfer of learning: Contemporary research and applications*. San Diego, CA: Academic Press.

Fabiani, M., Buckley, J., Gratton, G., Coles, M. G. H., & Donchin, E. (1989). The training of complex task performance. *Acta Psychologica, 71*, 259–299.

Fisk, A. D., & Gallini, J. K. (1989). Training consistent components of tasks: Developing an instructional system based on automatic/controlled processing principles. *Human Factors, 31*, 453–463.

Frederiksen, J. R., & White, B. Y. (1989). An approach to training based upon principled task composition. *Acta Psychologica, 71*, 89–146.

Gentner, D., & Stevens, A. L. (1983). *Mental models*. Hillsdale, NJ: Lawrence Erlbaum.

Gopher, D., Weil, M., & Siegel, D. (1989). Practice under changing priorities: An approach to the training of complex skills. *Acta Psychologica, 71*, 147–177.

Gropper, G. L. (1974). *Instructional strategies*. Englewood Cliffs, NJ: Educational Technology.

Jelsma, O. (1989). *Instructional control of transfer*. Enschede, The Netherlands: Bijlstra & Van Merriënboer.

Jelsma, O., & Bijlstra, J. P. (1988). Training for transfer in learning to detect, diagnose, and compensate system failures. *Proceedings of the Seventh European Annual Conference on Human Decision Making and Manual Control* (pp. 256–262), Paris.

Jelsma, O., & Bijlstra, J. P. (1990). PROCESS: Program for Research on Operator Control in an Experimental Simulated Setting. *IEEE Transactions on Systems, Man, and Cybernetics, 20*, 1221–1228.

Jelsma, O., Van Merriënboer, J. J. G., & Bijlstra, J. P. (1990). The ADAPT design model: Towards instructional control of transfer. *Instructional Science, 19*, 89–120.

Landa, L. N. (1983). The algo-heuristic theory of instruction. In C. M. Reigeluth (Ed.), *Instructional design theories and models* (pp. 163–211). Hillsdale, NJ: Lawrence Erlbaum.

Larkin, J., McDermott, J., Simon, D., & Simon, H. (1980). Models of competence in solving physics problems. *Cognitive Science, 4*, 317–348.

Lee, T. D., & Magill, R. A. (1985). Can forgetting facilitate skill acquisition? In D. Goodman, R. B. Wilberg, & I. M. Franks (Eds.), *Differing perspectives in motor learning, memory and control* (pp. 3–22). Amsterdam, The Netherlands: Elsevier Science Publishers.

Lewis, M. W., & Anderson, J. R. (1985). Discrimination of operator schemata in problem solving: Learning from examples. *Cognitive Psychology, 17*, 26–65.

Linn, M. C. (1985). The cognitive consequences of programming instruction in classrooms. *Educational Researcher, 14*(5), 14–29.

Mayer, R. E., & Greeno, J. G. (1972). Structural differences between learning outcomes produced by different instructional methods. *Journal of Educational Psychology, 63*, 165–173.

McDaniel, M. A., & Schlager, M. S. (1990). Discovery learning and transfer of problem-solving skill. *Cognition and Instruction, 7,* 129–159.

Merrill, M. D. (1983). Component display theory. In C. M. Reigeluth (Ed.), *Instructional-design theories and models: An overview of their current status* (pp. 278–333). Hillsdale, NJ: Lawrence Erlbaum.

Merrill, P. (1987). Job and task analysis. In R. M. Gagné (Ed.), *Instructional technology: Foundations* (pp. 141–173). Hillsdale, NJ: Lawrence Erlbaum.

Mettes, C. T. W., Pilot, A., & Roossink, H. J. (1981). Linking factual knowledge and procedural knowledge in solving science problems: A case study in a thermodynamics course. *Instructional Science, 10,* 333–361.

Morris, N. M., & Rouse, W. B. (1985). The effects of type of knowledge upon human problem solving in a process control task. *IEEE Transactions on Systems, Man, and Cybernetics, 15,* 698–707.

Myers, G. L., & Fisk, A. D. (1987). Training consistent task components: Application of automatic and controlled processing theory to industrial task training. *Human Factors, 29,* 255–268.

Olsen, S. E., & Rasmussen, J. (1989). *The reflective expert and the prenovice: Notes on skill-, rule-, and knowledge-based performance in the setting of instruction and training* (Tech. Rep.). Roskilde, Denmark: Risö National Laboratory.

Owen, E., & Sweller, J. (1985). What do students learn while solving mathematics problems? *Journal of Educational Psychology, 77,* 272–284.

Paas, F. G. W. C., & Van Merriënboer, J. J. G. (1992). Training voor transfer van statistische vaardigheden: Toepassing van een vier-componenten instructie-ontwerpmodel [Training for transfer of stastistical skills: Application of a four-component instructional design model]. *Tijdschrift voor Onderwijsresearch, 17,* 15–25.

Pea, R. D., & Kurland, M. (1984). On the cognitive effects of learning computer programming. *New Ideas in Psychology, 2,* 131–168.

Perkins, D. N., & Salomon, G. (1989). Are cognitive skills context-bound? *Educational Researcher, 18,* 16–25.

Pieters, J. M., Jelsma, O., & Van Merriënboer, J. J. G. (1987, September). *Skill acquisition: ADAPT instructional time to desired level of transfer.* Paper presented at the Second European Conference for Research on Learning and Instruction (EARLI), Tübingen, Germany.

Polson, P. G., & Kieras, D. E. (1984). A formal description of users' knowledge of how to operate a device and user complexity. *Behavior Research, Methods, Instruments, & Computers, 16,* 249–255.

Proctor, R. W., & Reeve, T. G. (1988). The acquisition of task-specific productions and modification of declarative representations in spatial-precueing tasks. *Journal of Experimental Psychology: General, 117,* 182–196.

Reigeluth, C. M. (Ed.) (1983a). *Instructional design theories and models: An overview of their current status.* Hillsdale, NJ: Lawrence Erlbaum.

Reigeluth, C. M. (1983b). Meaningfulness and instruction relating what is being learned to what a student knows. *Instructional Science, 12,* 197–218.

Reigeluth, C. M., & Merrill, M. D. (1984). *Extended task analysis procedures (ETAP): User's manual.* Lanham, MD: University Press of America.

Resnick, L. B. (1976). Task analysis in instructional design: Some cases from mathematics. In D. Klahr (Ed.), *Cognition and instruction* (pp. 51–80). Hillsdale, NJ: Lawrence Erlbaum.

Scandura, J. M. (1983). Instructional strategies based on the structural learning theory. In C. M. Reigeluth (Ed.), *Instructional design theories and models* (pp. 213–246). Hillsdale, NJ: Lawrence Erlbaum.

Schank, R. C., & Abelson, R. P. (1977). *Scripts, plans, goals, and understanding.* Hillsdale, NJ: Lawrence Erlbaum.

Schneider, W. (1985). Training high-performance skills: Fallacies and guidelines. *Human Factors, 27,* 285–300.

Schneider, W., & Fisk, A. D. (1982). Degree of consistent training: Improvements in search performance and automatic process development. *Perceptions & Psychophysics, 31,* 160–168.

Schneider, W., & Shiffrin, R. M. (1977). Controlled and automatic human information processing: I. Detection, search, and attention. *Psychological Review, 84,* 1–66.

Schoenfeld, A. H. (1979). Can heuristics be taught? In J. Lochhead & J. Clement (Eds.), *Cognitive process instruction* (pp. 315–338). Philadelphia: Franklin Institute Press.

Shea, J. B., & Zimny, S. T. (1983). Context effects in memory and learning movement information. In R. A. Magill (Ed.), *Memory and control of action* (pp. 345–366). Amsterdam, The Netherlands: Elsevier North-Holland.

Shepherd, A. (1986). Issues in the training of process operators. *International Journal of Industrial Ergonomics, 1,* 49–64.

Shepherd, A., Marshall, E. C., Turner, A., & Duncan, K. D. (1977). Diagnosis of plant failures from a control panel: A comparison of three training methods. *Ergonomics, 20,* 347–361.

Shiffrin, R. M., & Schneider, W. (1977). Controlled and automatic human information processing: II. Perceptual learning, automatic attending, and a general theory. *Psychological Review, 84,* 127–190.

Singley, M. K., & Anderson, J. R. (1988). A keystroke analysis of learning and transfer in text editing. *Human-Computer Interaction, 3,* 223–274.

Singley, M. K., & Anderson, J. R. (Eds.) (1989). *The transfer of cognitive skill.* Cambridge, MA: Harvard University Press.

Soloway, E. (1985). From problems to programs via plans: The content and structure of knowledge for introductory LISP programming. *Journal of Educational Computing Research, 1,* 157–172.

Sweller, J. (1988). Cognitive load during problem solving: Effects on learning. *Cognitive Science, 12,* 257–285.

Sweller, J. (1989). Cognitive technology: Some procedures for facilitating learning and problem solving in mathematics and science. *Journal of Educational Psychology, 4*, 457–466.

Sweller, J., Chandler, P., Tierney, P., & Cooper, M. (1990). Cognitive load as a factor in the structuring of technical material. *Journal of Experimental Psychology: General, 119*, 176–192.

Sweller, J., Mawer, R., & Ward, M. (1983). Development of expertise in mathematical problem solving. *Journal of Experimental Psychology: General, 112*, 634–656.

Tarmizi, R. A., & Sweller, J. (1988). Guidance during mathematical problem solving. *Journal of Educational Psychology, 80*, 424–436.

Tennyson, R. D., & Cocchiarella, M. J. (1986). An empirically based instructional design theory for teaching concepts. *Review of Educational Research, 56*, 40–71.

Tennyson, R. D., & Rasch, M. (1988). Linking cognitive learning theory to instructional prescriptions. *Instructional Science, 17*, 369–385.

Thorndyke, P. W., & Hayes-Roth, B. (1979). The use of schemata in the acquisition and transfer of knowledge. *Cognitive Psychology, 11*, 82–106.

Tromp, T. J. M. (1989). *The acquisition of expertise in computer programming skill.* Amsterdam, The Netherlands: Thesis Publishers.

van Merriënboer, J. J. G. (1990a). Strategies for programming instruction in high school: Program completion vs. program generation. *Journal of Educational Computing Research, 6*, 265–287.

van Merriënboer, J. J. G. (1990b). *Teaching introductory computer programming: A perspective from instructional technology.* Enschede, The Netherlands: Bijlstra & van Merriënboer.

van Merriënboer, J. J. G., & De Croock, M. B. M. (in press). Strategies for computer-based programming instruction: Program completion vs. program generation. *Journal of Educational Computing Research.*

van Merriënboer, J. J. G., & Krammer, H. P. M. (1987). Instructional strategies and tactics for the design of introductory computer programming courses in high school. *Instructional Science, 16*, 251–285.

van Merriënboer, J. J. G., & Krammer, H. P. M. (1990). The "completion strategy" in programming instruction: Theoretical and empirical support. In S. Dijkstra, B. H. M. Van Hout-Wolters, & P. C. Van der Sijde (Eds.), *Research on instruction* (pp. 45–61). Englewood Cliffs, NJ: Educational Technology.

van Merriënboer, J. J. G., & Paas, F. G. W. C. (1989). Automation and schema acquisition in learning elementary computer programming: Implications for the design of practice. *Computers in Human Behavior, 6*, 273–289.

Wightman, D. C., & Lintern, G. (1985). Part-task training for tracking and manual control. *Human Factors, 27*, 267–284.