Role of Instructional Theory in Authoring Effective and Efficient Learning Technologies†

Joseph M. Scandura

Scandura Intelligent Systems and
University of Pennsylvania

Abstract — This paper summarizes major issues in instructional theory and helps convey a better understanding of the advantages such theory offers for building intelligent instructional systems. Technology alone will not be sufficient for cost-effective development. Deeper understanding of the instructional process also will be essential. Taxonomic theory will not be sufficient for these purposes, however, nor will designer intuition. Good instructional theory must strive for comprehensiveness, internal consistency, parsimony and precision, and it must be operational. In this context, important advances have been made in instructional theory which hold considerable promise for the future. Building on core technology and the structural learning theory, a research program is proposed which will enable efficient development of effective learning systems for industry and schools. Copyright © 1996 Elsevier Science Ltd


Requests for reprints should be addressed to Joseph M. Scandura, University of Pennsylvania, c/o 1249 Greentree, Narberth, PA 19702. E-mail: jms@pobox.upenn.edu
Inadequacies in American education are well known. American students typically score near the bottom on international tests, especially in science and mathematics. Large segments of the American workforce lack the intellectual and technical skills necessary to compete in an increasingly competitive global economy. The problem is exacerbated by growing divisions in American society: some are well educated and prepared to compete with the best; others are ill-prepared to enter the workplace at any level — let alone to meet the increasingly demanding needs of an information-oriented society.

The potential benefits of learning technologies have been recognized for some time. In the 1960s, under the leadership of former Chairman William Norris, Control Data Corporation invested millions of dollars in computer-based instruction (CBI) on the bet that such technologies would revolutionize American education. Unfortunately, the costs were high and the results were mixed. Some good CBI programs were developed, undoubtedly, but mainframe computers used for delivery had major practical limitations, and the cost of producing CBI systems was exorbitant and the quality highly variable.

After 30 years of false starts, major advances in computer hardware and software technology — and increased understanding of the instructional process — have reached the point where a concerted R&D effort could provide the technology needed to meet the training and educational needs of the USA on a cost-effective basis. Over and above dramatic reductions in cost and increases in the power of computer hardware, new developments, for example, in simulation, multimedia, virtual reality, and similar technologies, have reached the point where it is possible to create environments in which learners can more efficiently acquire the knowledge and skills that they need in a highly efficient environment. While ongoing development of these technologies will increase the scope of what might be accomplished, these technologies alone will not be sufficient.

Coordinating technologies will, in many ways, be even more important. Just as coordinating movements of mechanical arms, vision technology, etc., is crucial in producing working robots, coordination will be essential if we are to build quality learning systems in a cost-effective way.

The main point of this white paper is that instructional theory can provide the necessary coordination in learning technology. Indeed, important advances have been made in instructional theory — advances which hold considerable promise for the future. While recognizing that supporting technology will be essential to the success of any commercially viable application of learning technology, I concentrate here on the equally crucial role of instructional theory in making such efforts successful. In particular, the immediate goal of the proposed program is to develop core technology
enabling the cost-effective development of effective and efficient learning environments for industry and schools.

Currently, the only type of instructional theory used in designing learning environments (where such theory is used at all) is primarily taxonomic. Different approaches are typically used to achieve different learning outcomes — one method for facts, say, and another for teaching problem solving. Intelligent tutoring systems also tend to be atheoretical. Reliance is typically placed on the intuition of the designer. We must do better if we are to achieve the aforementioned goals.

I believe that instructional theory must be practical. However, I am of the old school insofar as theory is concerned. Good instructional theory is not a compendium of diverse theoretical perspectives, nor is it constructive philosophy or any set of techniques or methods or intuitions. Instructional theory should strive for comprehensiveness, internal consistency, parsimony, and precision, and it must be operational.

Do we have such a theory today? Not entirely, but the Structural Learning Theory provides a sound foundation on which to build. My own work in this area goes back many years and is summarized in Figure 1.

In this paper, I will not dwell on theoretical details — nor is that necessary since much of the material is available in published form. Instead, I hope to summarize some major concerns, and to convey both a better understanding of instructional theory and the advantages that such theory offers for building sound instructional systems.

**BACKGROUND**

The basic datum in studying human cognition is human behavior. As educators and psychologists, we do not have the luxury of looking inside the brain. Although there is unquestionably a good deal to learn about the brain, I dare say that we will never be able to teach higher mathematics or poetry, for example, by programming human gray matter. Of necessity, we work at a higher level.

In short, we cannot hope to get the information we need to create effective learning situations by simply studying the brain. Nor can we directly manipulate the brain of a child. We can only infer things about the mind by observing behavior.

That, of course, is the problem. Exactly how do we find out what a student does and does not know by observing behavior? Traditional cognitive psychology tackles this problem by building models, minitheories if you will, about how people solve this or that problem, or perform this or that task. These models, in turn, lead to behavioral predictions. Often, these
Figure 1. An overview of major phases of Scandura's research arranged chronologically. Captions in the rectangles refer to key results. Relationships are represented by arrows between the rectangles. S–R = stimulus–response; SLT = Structural Learning Theory; T–L = teacher–learning.
predictions map reasonably well onto behavior. Yet, they typically say little about individual behavior. Although consistent with behavior averaged over groups of people, these models may differ significantly from what individuals do and do not know. To address individual behavior might require as many models as there are individuals, clearly an impractical solution. To make matters worse, different theories are needed for different content domains.

STRUCTURAL ANALYSIS

Clearly, fundamental instructional theory requires a different approach (Scandura, 1971c, 1973b; Scandura, Durnin, & Wulfeck, 1974a). Rather than developing each content-specific theory de novo, a common method, metatheory if you will, is needed for constructing content-specific theories. Task analysis provides a case in point. It has been used to analyze a wide variety of task domains. Sharing much in common with structured analysis in software engineering, task analysis identifies the ingredients (or prerequisites) for performance at successive levels of abstraction. What it does not do, however, is specify cognitive processes sufficient for actually solving problems in the domain.

Structural analysis is a form of cognitive task analysis which explicitly provides for higher as well as lower order knowledge (Scandura, 1973b, 1984c; Scandura et al., 1974a). Given an arbitrary content domain, no matter how simple or how complex, structural analysis results in a set of rules and higher order rules which collectively make it possible to solve a wide variety of tasks in the domain — including novel tasks not directly considered in the analysis.

Structural analysis provides a different approach to the question of individual knowledge, one based on the fact that we can never know everything that any person knows. Nor is that necessary. We need only know what an individual knows relative to what we are interested in observing. Furthermore, what we are interested in observing can always be represented in terms of rules and higher order rules identified via structural analysis. These rules provide a prototype, a measuring device of sorts, against which to measure the knowledge of individuals.

In effect, although we can never know exactly what a person knows, we can determine which parts of which rules are and are not known. Moreover, extensive research shows that this information can be used to predict individual behavior with an unusually high degree of accuracy.

In this context, let me emphasize a few points about structural analysis that are often misunderstood.
1. Structural analysis is strictly cumulative. Suppose one enlarges the domain of analysis, for example, by extending an analysis of 'ing' endings to a broader range of grammar, or by generalizing computational arithmetic to incorporate concrete meanings as well. Do we have to 'throw away' the initial results (of structural analysis), that is, must we start over, by applying structural analysis to the enlarged domain? Happily, the answer is no. Structural analysis simply begins where the previous analysis leaves off (Scandura, 1977b, especially chapters 3, 13, 14).

2. Structural analysis is applicable to essentially any content, no matter how broad and comprehensive (Scandura, 1973b, 1977b, 1981a) or how narrow and focused (Scandura, 1981a; Scandura & Scandura, 1980). Ironically, full analyses of both kinds of domains invariably result in higher order rules. In the case of broad domains, the higher order rules typically serve to create (i.e., generate) entirely new rules (Scandura, 1973b, 1977b). In the case of highly specific knowledge, the higher order rules commonly involve automatization (Scandura, 1981a; Scandura & Scandura, 1980). These higher order rules replace procedural complexity with structural complexity and, consequently, result in more efficient rules.

3. Structural analysis may be applied to a given content domain from any number of different perspectives. Each perspective corresponds to a different paradigm for addressing that content. The borrowing and equal additions methods of column subtraction provide a simple example (Scandura & Durnin, 1973). Pragmatic versus theory-based approaches to instructional design provide others. Yes, instructional design itself can be subjected to structural analysis.

   Each perspective corresponds to a common view shared by individuals in some group or subgroup. The rules associated with each such perspective best characterize the knowledge of individuals in the corresponding group.

   This observation raises a variety of issues. Communication, for example, may be difficult if not impossible where teacher and learner — or any two individuals for that matter — approach a problem domain from different perspectives. To communicate, the two must first establish common ground. This requires common elements in the results of structural analysis from the opposing perspectives. This is an area full of research opportunities.

4. It is not essential that the results of structural analysis capture individual knowledge exactly. It is sufficient only that the identified rules yield (equivalent) predictions. Given a problem domain, there always exists some level of analysis beyond which predictions cannot be distinguished.
If accurate computation is all that matters from an instructional design perspective, for example, a more detailed analysis involving processing time would be superfluous. The former analysis might be quite sufficient. Having said this, I would add a caveat. Just because an analysis (i.e., rule set) cannot be distinguished within a defined domain does not mean that it cannot be distinguished. Generalizing or enhancing a domain (e.g., by introducing response times), for example, makes it possible to distinguish behaviorally between different levels of analysis (Scandura, 1973b, 1977b).

5. Detailed structural analysis of nontrivial domains can take a lot of work. It is not essential, however, to detail everything. Experience shows that even crude structural analyses can have valuable predictive benefits (Scandura, 1977b; Scandura & Scandura, 1980). Although deterministic predictions are theoretically possible, this is not essential. The question is whether structural analysis, formal or otherwise, adds a degree of predictability that would not otherwise have been possible.¹

Rules play a central role in most well-formed learning theories. Let me be a bit more precise on how they are formulated in the Structural Learning Theory. All rules have a domain, a range, and an operation or process connecting the two. The domain may consist of any number of elements, and similarly for the range. The range of a rule, however, is not necessarily what one gets by applying the operation to elements in the domain. Rather, the range of a rule defines the expectations of applying the rule. Rule domains and ranges, together with a universal content-independent ‘goal switching’ control mechanism, play a central role in determining which rules will be used in given problem situations (Scandura, 1971b, 1973b, 1981a).

Higher order rules are simply rules in which the elements of the domain and/or range may themselves be rules. All higher order rules combine, or otherwise modify, given rules in one way or another. None the less, it is often convenient to distinguish two kinds of higher order rules. Some higher order rules produce new knowledge: for example, generalizations of existing rules or new combinations of such rules (Scandura, 1971b, 1973b, 1974, 1977b). These are the kinds of rules that are used in learning or problem-solving situations where the student does not already know how to deal with given problems.

Other higher order rules, however, result in automatization. Such higher order rules make existing rules more efficient (Scandura, 1981a; Scandura & Scandura, 1980). They convert operational steps in the procedural portion of a rule into automatic processes encapsulated in domain or range objects. The terms ‘objects’ and ‘encapsulation’ are used purposefully in this context to emphasize a very strong analogy between what I am proposing and the
object-oriented paradigm currently in vogue in the software engineering community (Scandura, 1994a, 1996). When applied to the design of software systems, structural analysis constitutes a new cognitive paradigm which effectively generalizes and incorporates the traditional object-oriented paradigm. However, let me not diverge.

All this is well and good — but it still does not deal with the question of how to account for creative or problem-solving behavior. One possible solution derives from a very simple idea. Recall the chicken in your Psychology I textbook placed in front of a glass barrier separating it from its food. First, the chicken bangs into the glass. After crashing enough times, the chicken by trial and error may eventually find its way around the barrier. While I am not inclined to attribute high intelligence to the typical hen, or rooster for that matter, trial and error is just the name given to a class of higher order processes for deriving solutions for unfamiliar problems. Indeed, such behavior appears universal. When faced with a problem for which one does not have a ready-made solution process, humans — and apparently chickens as well — set about the task of finding a solution that will work (Scandura, 1971b, 1973b, 1974, 1977b).

When thus formulated, what I have called the ‘goal switching’ mechanism will not work with arbitrary knowledge in a generalizable way. In its original form, goal switching subtly but necessarily required content-specific knowledge in addition to the mechanism itself (Scandura, 1977b, 1981a). There is, however, a solution to the problem, one which, based on my other experiences with the US Patent Office, appears to be patentable. In short, there is far more to this simple-minded idea than first meets the eye.

OVERVIEW OF INSTRUCTIONAL THEORY

Let me now turn specifically to instructional theory (Scandura, 1981b, 1987b, 1987c; Scandura & Scandura, 1987; Scandura, Stone, & Scandura, 1986). Figure 2 shows how relationships are handled in the Structural Learning Theory between content, the idealized competence used to represent that content, the teacher and the learner. Formal representation of the content (idealized competence) is obtained via structural analysis. The teacher and learner communicate through that representation. For example, this representation serves as a criterion against which the teacher measures the learner’s knowledge (relative to what the teacher wants the learner to know). The learner (model) is characterized in terms of both universals and a relativistic characterization of the domain as formally represented (in terms of rules and higher order rules). Notice in Figure 2 that the teacher and learner necessarily communicate through — or with respect to — one or
Figure 2. Schematic showing how relationships are handled in Structural Learning Theory.

more perspectives on a body of content. These perspectives consist of sets of rules, some of which may operate on other rules.
The teacher normally is assumed to have full knowledge of the rules and higher order rules constituting the target content domain. Only the learner’s knowledge is at question. This knowledge is assessed relative to rules in the content domain (Scandura, 1987c; Scandura & Scandura, 1987). Theoretical problems become more complex when both teacher and learner have only partial knowledge.

I will not attempt here to detail exactly how the teacher decides what problems to present to the learner, how the teacher assesses what the learner knows and/or how the teacher determines what information to present to the learner at each point in time. For this, I refer you to the listed publications (Scandura, 1987c; Scandura & Scandura, 1987).

I would, however, emphasize one important point. This teaching-learning paradigm is applicable not only to procedural knowledge. The theory applies to all kinds of knowledge whether that involves learning simple facts, learning how to apply general purpose solution rules (such as those used in solving subtraction problems), or using existing knowledge (rules) to derive new rules for solving novel problems.

Based on earlier research, it is fairly clear how the latter two forms of learning can be handled. On deeper analysis, one also finds that learning simple facts follows the same principles. What may appear to be simple facts, almost always have a complex genesis. In analyses of Piagetian conservation, for example, what appear to be automatic conserver responses are derived from more basic knowledge via automatization (Scandura & Scandura, 1980). Even S–R learning, involving well-established and familiar stimulus and response objects, follows the same pattern once context is taken into account. In traditional S–R research, for example, each S–R association is embedded in a list (‘situational cognition’ perhaps?).

**IMPLICATIONS FOR LEARNING TECHNOLOGIES**

Suppose for a moment that we could develop a general purpose intelligent tutoring system, and that this tutoring system could accommodate essentially any type of subject matter (Scandura, 1987c; Scandura & Scandura, 1987). Unlike current tutoring systems, the system I have in mind would automatically generate test problems as needed. It would automatically infer what the learner did and did not know. Also, it would provide precisely the instruction each learner needed when it was needed. It is important to distinguish this type of tutoring system — both from traditional authoring systems, which although ‘general purpose’ require designer input for instructional specifications as well as content, and from intelligent tutoring
systems, which although ‘intelligent’ find content and instructional decision making inextricably intertwined.

Imagine next what happens as the teacher knows less and less about the content to be taught. The interaction gradually degrades. Eventually, the teacher may perform comparably to the learner. In this case, we have a model for cooperative student learning.

Next, suppose that the teacher knows and contributes nothing to the dialogue. Here, the learner is interacting with an essentially random environment centered on the content domain. The learner finds himself exploring the content in an open-ended simulation environment.

In all of these cases, our general purpose tutor would relieve us of a good deal of work. The construction of test items, their answers and instruction, as well as the sequencing of testing and instruction would all be handled automatically. In the case of simulation, of course, the learner alone would be the focus of interactivity.

In order to work as a fully fledged system, the general purpose tutor needs content. More particularly, it needs sets of rules which characterize the problem domain in question. This requires structural analysis. Indeed, in this context, instructional design actually reduces to structural analysis. Once built, the tutor would take care of everything else: devising test problems, sequencing of tests and instruction, even deciding how information is to be presented.

Structural analysis, of course, does require thought and effort. The tutoring system, however, would release you, as an instructional designer, of essentially all instructional decisions. These would be fully automated. In comparison with traditional authoring systems (whether developed by us, e.g., Scandura, 1983, or others), the savings would be considerable. There would be no need to worry about instructional decision making — about which or how many test items to use to assess student knowledge or about what instruction to give and when. Moreover, the results would be far more predictable. If we have done our job right — our goal would be to optimize the teaching—learning process — student learning would be far more efficient than if instruction were designed by a human. In this context, our analyses suggest that human teachers simply could not attend to the detail required for optimized instruction (Scandura, 1987c; Scandura & Scandura, 1987).

Of course, I am not so naive as to believe that we can easily develop such a tutoring system — one that will accommodate all varieties of content. Having supervised the development of some very large (million-line plus) systems in software engineering, I am well aware of the technical issues that will have to be resolved. However, that is where theory comes in. Sound instructional theory can provide a beacon to guide development. It can lead
to powerful, highly efficient and truly general purpose intelligent tutoring systems. It can ensure development that is cumulative rather than haphazard. More generally, sound theory can save money and a lot of wasted effort.

In many ways even more important, we have developed enabling technologies in software engineering which for the first time make it possible (i.e., feasible) to implement a Generalizable Author-Trainer system based on the Structural Learning Theory. These technologies are based on a new cognitive analysis, design and programming paradigm which is the natural culmination of our work in structural analysis. This paradigm represents a major generalization of the popular Object-Oriented (OO) paradigm. We call it the Cognitive Object-Oriented (COO) paradigm (e.g., Scandura, 1994b). The COO paradigm makes it easier to represent 'real world' models in semantically meaningful terms and solves many limitations of the OO paradigm.

Figure 3 shows how the 'flexsys' designer, simulator and semantic components will be used for creating intelligent tutoring systems in which content and instructional expertise are completely separate. The GAT Author will be used to construct executable representations (in terms of rules and higher order rules) of the content to be learned. The GAT Trainer Development System, using a somewhat broader range of flexsys compo-
nents, will be used to develop GAT Trainers for delivering instruction. Our flexsys tool set (which supports the COO paradigm) includes a number of components, most importantly a major breakthrough in getting software components from essentially any source to work together seamlessly, based solely on semantic considerations. Use of these technologies would significantly reduce the cost of building intelligent tutoring systems of the type proposed earlier. The Structural Learning Theory, in turn, provides the guidance necessary to optimize instructional efficiency — in ways which in most cases would exceed human capacity. Theory and reduced development costs will both be essential if sound instructional systems (as opposed to ‘Edutainment’) are to become commercially viable.

NOTE

1This does not necessarily imply a need for traditional statistical methods. Indeed, so-called ‘fuzzy sets’ could well provide a better model where content does not readily lend itself to sharp partitioning.

REFERENCES


