11

DNA: Toward an Automated Knowledge Elicitation and Organization Tool

Valerie J. Shute, Lisa A. Torreano, and Ross E. Willis
GKIS, Inc.

The research described in the first volume of *Computers as Cognitive Tools* reflected the premise that computer power can be harnessed in multifarious ways to enhance student learning. The editors described this research as falling along a dichotomy between two camps—modelers and nonmodelers—with a third camp bridging the two. Regarding this dichotomy, then and now, our tent is pitched in the modelers' camp. This view holds that modeling the learner renders computer-based instruction more intelligent, and thus more effective (Anderson, 1993; Shute & Psotka, 1996). However, whereas the previous volume used modeling to denote the process of representing students' knowledge structures, we expand the term to include the process of representing the domain or task being instructed. Modeling in this context allows the computer to know what to teach, as well as when and how to teach it.

There are three agreed-on components that serve to make computer-assisted instruction intelligent: an expert model, a student model, and an instructor model (Lajoie & Derry, 1993; Polson & Richardson, 1988; Psotka, Massey, & Mutter, 1988; Shute & Psotka, 1996; Sleeman & Brown, 1982). Basically, the expert model represents the material that is to be instructed—the ideal representation of the domain or task. In essence, it is a blueprint of the knowledge elements and their associated structure and interdependencies. The student model represents the student's knowledge and progress in relation to this blueprint. Finally, the instructor model customizes the instructional experience for each learner based on discrepancies between the student and expert models. This is achieved by embodying theories of learning that guide the course of instruction in the program.

This chapter describes the new computer program Decompose, Network, Assess (DNA). We discuss it in conjunction with another system, called Student Modeling Approach for Responsive Tutoring (SMART; Shute, 1995), because both attempt to render computerized instructional programs intelligent. The programs work in concert, such that DNA extracts and organizes knowledge and skills from subject matter experts and SMART uses the resulting structured curriculum elements as the basis for assessment, cognitive diagnosis, and instruction. In other words, DNA provides the blueprint for instruction, obtaining curriculum elements directly from the responses and actions of multiple subject matter experts, who answer structured queries posed by the computer (Shute, Willis, & Torreano, 1998). The student modeling paradigm (SMART) assesses performance on each curriculum element by way of a series of regression equations that are based on the level of assistance the computer gives each person per element (Shute, 1995). Thus, DNA relates to the "what" to teach, and SMART addresses the "when" and "how" to teach it.

Historically, specifying what to teach has hampered efforts to develop intelligent instructional software efficiently. In fact, due to its time and resource costs, it has often been referred to as the bottleneck in the development process (Durkin, 1994; Gordon, Schmierer, & Gill, 1993; Hayes-Roth, Waterman, & Lenat, 1983). That is, the processes of eliciting and hierarchically organizing the necessary elements for an expert model involve exorbitant amounts of time to accomplish, and even then are more art than science. Despite the fact that the expert model is difficult to develop, it is often characterized as the backbone of any intelligent instructional system (Anderson, 1988). Therefore, our aim with DNA is to attempt to open up this bottleneck. We wish to increase the efficiency of developing the expert model by automating the bulk of the knowledge elicitation and organization processes. This automated approach to creating the expert model is embodied in DNA.

FOUNDATIONS OF DNA

We begin this section with an overview of the SMART framework—presented to highlight the content and structure requirements for the DNA program. Relevant cognitive analysis techniques are discussed.

The SMART Framework: Precursor to DNA

Three basic features of SMART directly influenced DNA design decisions. First, SMART requires the categorization of each bit of knowledge or skill, comprising some domain, into one of three different learning outcome categories: symbolic knowledge (SK), procedural skill (PS), and conceptual knowledge (CK). Before DNA was developed, several independent raters achieved this categorization of elements by applying well-defined operational definitions. The simplified operational definitions are: SK: knowledge of any symbol, formula, basic definition, or rule; PS: the application of a formula or rule, or performing a specific action within the tutor; and CK: the definitions of, and relations among, various concepts. Basically, this represents a slight extension of the well-established declarative-procedural knowledge distinction (see Anderson, 1983, 1993).

Second, SMART differentially instructs curriculum elements (CEs) based on these outcome types. For instance, symbolic knowledge is instructed by means of drill and practice. Procedural skill is instructed by presenting problems to solve that are specifically related to either the CEs that are currently being instructed or the CEs that were inferred as the bug in the learner's knowledge and therefore require remediation. Finally, conceptual knowledge is instructed by carefully designed analogies (Shute, 1994, 1995). This attempts to capitalize on the best aspects of a variety of theoretically grounded student modeling approaches by pairing each approach (drill and practice, problem solving, or analogies) with the most appropriate knowledge or skill type (Shute & Catrambone, 1996). Thus, instruction methods are applied differentially to distinct knowledge types to optimize learning.

Third, SMART relies on the inheritance relationship of a hierarchical structure of CEs for managing assessment and instruction. That is, the underlying knowledge base consists of CEs arrayed such that their relationships are clarified. The hierarchical structure denotes elements that are basic or prerequisite to more complex bits of knowledge. This influences instruction and assessment in that more basic, prerequisite knowledge elements are instructed prior to more complex dependent ones, and deficiencies in learner performance are inferred based on these dependency relations. For instance, one must know the individual symbols of Σ, X, and N before understanding the formula for the mean: $(\Sigma X)/N$. Therefore, these symbols would be instructed prior to the formula for the mean. In addition, if the learner's knowledge of the formula for the mean is deficient, then the hierarchical structure of CEs indicates which knowledge elements may be the source of the deficiency and therefore deserve remediation. Structurally and functionally, this knowledge structure constitutes a learning hierarchy (Gagné & Briggs, 1965).

These three basic features provide the instructional framework of SMART and define the parameters and criteria for DNA's design. Relying on SMART's framework is justified because the efficacy of this approach has been empirically validated. That is, a controlled evaluation examined learning gains between participants using one of two versions of the same tutor: with and without SMART enabled. Findings showed that learners in the non-SMART version showed impressive learning outcome scores (2 standard deviation pretest to post-test improvement). Their final post-test scores were 74.9% on average. Learners in the SMART version showed even higher gain scores: average post-test scores of 82.1%. An analysis of covariance was computed on the post-test data with pretest as a covariate and version as a between-subjects variable. Results showed a significant difference in learning outcome due to version: $F(1, 199) = 4.16$; $p < .05$, with superior outcome performance evidenced by participants in the SMART-enabled condition (Shute, 1995).

In summary, the empirical success with SMART has motivated key DNA design decisions. Specifically, we decided to require DNA to elicit and structure information so that it fits SMART's database requirements of three outcome types: SK, PS,[1] and CK. This categorization scheme allows for the analysis of a wide array of domains or tasks, rendering DNA a general-purpose tool for specifying curriculum. To accomplish this, DNA asks subject matter experts a semistructured series of what, how, and why questions—the analogues to symbolic, procedural, and conceptual knowledge. In addition, the success of the hierarchical structure of SMART's underlying knowledge base resulted in our decision to include a separate module in DNA to obtain the spatial and conceptual organization of elements needed for a sound curriculum.

Knowledge Elicitation and Organization Techniques

What is demanded of the methods used to conduct a cognitive task analysis (CTA) is jointly determined by the purpose of doing the analysis and the type of domain or topic that is to be analyzed. These two critical factors determine what is required of a useful and appropriate knowledge structure. Traditionally the primary purpose for conducting a CTA has been to delineate an expert's performance in relation to some task, down to a fairly small grain size (e.g., elementary cognitive processes). However, given our specific interest in developing curriculum for intelligent instructional systems across a broad spectrum of topics, the analysis techniques we include in DNA must be able to apply to both domains that involve performance of a task and those that do not. These requirements guided the choice of which techniques would be appropriate to embed in DNA. Due to our goal of broad applicability of the tool, we use "cognitive task analysis" to denote any systematic decomposition of a domain in terms of constituent knowledge and skill elements.

[1]Procedural knowledge (PK) is another outcome type, but it is subsumed under symbolic knowledge (SK), which can be divided into simple and complex components (for more on these knowledge types, see Shute, 1995).

Knowledge Elicitation. Interviews constitute a fundamental method for eliciting information from experts. The nature of the interview is typically based on a theory of expertise and is designed to fit the framework of the purpose for which the cognitive task analysis is being conducted (Ryder & Redding, 1993). In other words, the form the questions take and the order in which they are posed can vary according to the information one wishes to elicit. Interview methods can be structured or unstructured and can be concurrent or retrospective with the performance of a task being analyzed.

Our purpose for conducting cognitive task analysis is to obtain ample data on some topic or task for instructional purposes. The virtue of interview techniques lies in their flexibility and directness; thus, they can be used to analyze a wide range of topics which suit our particular goals. To obtain such data, appropriate questions embodied within the interview should probe the expert for as much information as possible per curriculum element. For instance, for procedural topics, experts should be asked to specify what actions and steps are relevant, how they are best accomplished, and why those steps are taken instead of alternative ones. For more conceptual issues, experts should be asked to specify what defining traits and examples are important, how they are related to the concept, and why they are consequential.

Knowledge Organization. After information from an expert is obtained, how is it optimally represented or arrayed? Conceptual graphs are one popular means of representing hierarchically-structured knowledge. As the name implies, conceptual graphs are the graphical representation of concepts showing, at various grain sizes, relevant concepts (nodes) and their interrelationships (arcs or links). This representation format resembles semantic networks in cognitive psychology (Collins & Quillian, 1969; Jonassen, Beissner, & Yacci, 1993). One beneficial characteristic of this type of representation is that it depicts information in such a way that allows inferences to be made. That is, the hierarchical structure between nodes provides "inheritance" information: a subordinate (or "child") node inherits the properties of its superordinate (or "parent") node. For example, if a canary is specified as a "kind of" bird, then it can be inferred that the properties and characteristics associated with "bird" also apply to the concept of canary.

Another popular form of knowledge representation is a production system framework that results from a GOMS-type analysis (Goals, Operators, Methods and Selection rules; Card, Moran, & Newell, 1983). Again, as the name suggests, this representation specifies the goals that are to be achieved, the methods or steps taken to achieve those goals, and the criterion on which alternative steps are selected. One beneficial characteristic of this representation is that it has the potential to be easily translated into an executable system. That is, condition-action pairs define what must occur in order for some action to fire (see Anderson, 1993; Gray, John, & Atwood, 1993; Newell, 1990).

Summary. These knowledge elicitation and organization techniques have proven helpful when used for their respectively appropriate purposes and topics of analysis. Many other techniques exist and can be used collaboratively to balance each method's strengths and weaknesses. Successful elicitation techniques include document analysis, observation of experts, and protocol analysis that requires experts to "talk aloud," voicing their mental processes while performing the target task. In addition, techniques such as card sorting, ordered recall, similarity judgments, ranking, and ratings are useful techniques for eliciting and structuring knowledge. However, for our current purpose of putting the design decisions of DNA in context, we will not address these other techniques. (For good reviews of CTA methods, see Schraagen et al., 1997, and Williams, 1993. For a fuller discussion of knowledge organization and representation issues, see Jonnassen & Carr, this volume.)

DESCRIPTION OF DNA

The primary goals of DNA are twofold: to maximize the range of domains that can be analyzed with a single CTA method and to optimize the cost-benefit ratio of the process. As a bonus, DNA is intended to provide a principled approach to the currently unstandardized process of knowledge elicitation and organization.

We view CTA as any systematic decomposition of a domain in terms of constituent knowledge and skill elements, whether the domain is related to task performance (e.g., troubleshooting jet engine malfunctions) or not (e.g., understanding the core concepts of religions). Therefore, to achieve the first goal's capability of this breadth of knowledge representation, we chose to create and employ a hybrid output structure involving a mixture of semantic net and production system architectures.

To optimize the cost-benefit ratio of doing cognitive task analysis, DNA is automated. The intention is to improve efficiency by decreasing the personnel resources (and, hence, time and cost) required in the analysis. Traditional CTA consists of two distinct phases: elicitation of knowledge and skills and the organization of those elements. These phases customarily occur at different points in time and often with different persons doing the elicitation and organization. For example, a knowledge engineer interviews or observes a subject matter expert (SME), while a cognitive psychologist or instructional designer takes the output and arranges it into a conceptual graph or production system. With DNA, these two phases are collapsed into a symbiotic process in order to decrease the time and cost associated with conducting two separate analyses. Thus, in DNA, the SME identifies all CEs and arrays them in a hierarchical structure.

Modules of DNA

DNA consists of a core of four interactive modules that automate the knowledge elicitation and organization processes: Customize, Decompose, Network, and Assess. The Decompose module was designed to be a running dialogue between the computer and the SME. It works by asking structured interview questions while an expert decomposes a domain using keyboard input. After decomposing a domain into individual curriculum elements, the SME networks the elements into a learning hierarchy (Gagné & Briggs, 1965). Finally, the SME's data are assessed for validity (i.e., accuracy and completeness) by distributing his or her learning hierarchy to other SMEs, who edit its structure and content.

Customize. In DNA's Customize module, the instructional designer (ID) provides information about which domain is to be decomposed. This information then goes to the SMEs. In particular, the ID specifies the ulti-

Dear [Name of Expert], [Today's Date]

We're writing today to get your help in designing a course on Microsoft Exchange (ver. 4.0). Before you begin working with the enclosed DNA program, please sit down and think about the critical things that make you good at using the MS Exchange software package.

As you go through DNA and respond to our questions, try to respond in terms of how you currently perform the job or think about the particular task. Please do not respond with how you originally learned Microsoft Exchange; you have probably developed much better ways of performing this task since then.

The ultimate goals of the course are for our students to:
(1) Know how to create a new email message
(2) Know how to address email
(3) Know how to send email

How specific should you get? You can presume that our students will have the following knowledge and skills:

(1) Knowledge and skills with Windows 95
(2) Basic word processing skills

Therefore, you will not need to define knowledge or skills at a detailed level in relation to these elements.

When answering questions during the program, please adjust your responses to fit the following guidelines:

What box: 10%
How box: 85%
Why box: 5%

Thanks very much for your time.

Sincerely,

FIG. 11.1.

mate learning goal of the tutor to be developed, prerequisite knowledge and skills of the learners, and the desired instructional emphasis or flavor (e.g., primarily procedural). This information provides the SME with the superordinate goal of the analysis and the lowest level subordinate goal, at which point the SME should stop decomposing the domain. Using the information provided by the instructional designer, the Customize module generates a personalized letter explaining the purpose of the project (which the ID can edit) and a set of floppy diskettes that will be mailed to prospective SMEs. The diskettes contain files for a SME to install on the computer that DNA needs to elicit and store knowledge structures.

For example, Fig. 11.1 shows a letter that was generated by the Customize module and used in a formative evaluation of the system and some in-house SMEs analyzing the domain of Microsoft Exchange (mail software). Upon receipt of the letter and installation of the software, the SME goes through a short (10 minutes) orientation program that provides an overview of DNA and transitions directly into the Decompose module.

Decompose. The Decompose module consists of a semistructured, interactive dialogue between the computer and the SME. It was designed to elicit most of the explicit knowledge associated with the domain or topic of analysis. DNA uses a series of three interrogation branches to elicit knowledge from experts. The symbolic (or "what") branch elicits SK by asking experts to provide definitions of terms used in his field of expertise. The procedural (or "how") branch elicits PS by asking experts to outline specific steps, conditionals, relational connections, and subprocedures of a procedure. While responding to questions in the procedural branch, experts may also provide SK elements by defining ambiguous terms and attaching multimedia files, such as pictures, movies, and sounds. The conceptual (or "why") branch elicits CK by asking experts to delineate the important components in their domain and explain how these components are functionally related. Additional CK is derived from experts who are asked to specify their understanding of why these components are important in relation to the overall learning goal. In general, DNA utilizes the "what, how, and why" questioning procedure that has been shown to elicit knowledge from experts successfully (Gordon et al., 1993).

The questioning sequence is left to some degree to the discretion of the SME, who is allowed to decide which main question to answer. This enables the expert to decompose the domain in a breadth- or depth-first manner. For instance, an expert can begin by generating a number of higher-level goals, then proceed to describe these goals at a more specific level across the topics (breadth first). Alternatively, the expert can start by identifying a single high-level goal and then delineate its lineage (depth first). Low-level, or terminal, nodes are determined by the description of learners' incoming knowledge and skills, specified in the customized letter. This flexibility differs from more rigid cognitive task analysis approaches, like GOMS, which force an expert to decompose a domain in a breadth-first manner (Williams, 1993).

During the Decompose module, all information about evolving curriculum elements is stored in a Microsoft Access 7.0 database. Each CE receives a unique number (assigned by DNA), as well as a name and description (provided by the SME). The numbering system reflects the order in which the CEs were specified by the SME and inherently contains information about higher order relations. That is, each element is given a unique number that designates it as a main element, a step within a procedure (or subprocedure), or a definition associated with either a step or a main element. Main CEs are given a unique integer, and steps within a procedure are given a number based on the main CE number and the number of the step. For example, the third step associated with CE number 2 would be numbered 2.003. Substep numbering follows the same logic but goes out three more decimal places. This numbering scheme allows a procedure to have up to 999 individual steps. Definitions associated with steps are given the main CE number, the step number, substep numbers, and the number of the definition. For example, the ninth definition associated with step number 3 for CE number 2 would be labeled 2.003.09. This numbering system allows for 99 definitions to be associated with each step or substep.

The CE name and description are obtained directly from the SME. Additionally, the program categorizes each CE in terms of knowledge type (SK, PS, or CK) based on the type of question that elicited the CE. For example, if the SME created a new CE in response to a "What do you typically do first when you . . . ?" query, that would be classified as PS. Finally, the expert can also attach graphical or sound files to individual CEs for greater elaboration and clarification. Together these data will enable the instructional designer to develop a rich and sound curriculum.

Extensive help, either solicited by the SME or provided by the system (on a need basis), is available throughout the Decompose module. In addition to the orientation, there are many online help tools. For instance, information is always available to the SME in the form of examples and pointers. Examples of valid entries are presented from a variety of domains, as are answering pointers that guide the wording of the SME's input. Furthermore, a help screen appears for the SME whenever an interface or decision point is first encountered.

After delineating all elements related to the domain, the expert proceeds to the Network tutorial, which gives the SME a basic understanding of hierarchical structures and provides practice in arranging curriculum elements that are represented as nodes. That is, the expert is instructed on how to build a graphical representation by arranging practice nodes in a hierarchical manner, much like a conceptual graph (Gordon et al., 1993). This tutorial also teaches experts how to create new nodes and link existing ones together. This work leads directly into the Network module.

Network. The Network module (currently under development) is intended to transform curriculum elements elicited during the Decompose module into graphical nodes that experts spatially arrange and link to represent hierarchical conceptual graphs or production rules. Each node contains the name of the CE as defined during the Decompose module. Node shapes differ by SMART's learning outcome type: Rectangles represent SK, ovals are PS elements, and rounded rectangles denote CK.

To simplify viewing and editing, only main-level CEs and their first-level "children" (nodes) appear on the initial screen. "Pregnant" CEs are those that have elements embedded within them. They appear in bold type. Any pregnant element can be unpacked to expose its constituent parts by right clicking on the node and choosing the option Unpack.

Some links are already in place when the SME arrives at the Network module. These come from information provided during the Decompose module, such as higher order relations inherent in the decomposition of a procedure. Other links must be drawn and labeled. CE links may vary along three different dimensions: type, directionality, and strength of association.

The first kind of link relationship is *type*. These denote the specific kind of relationship between nodes. DNA's semantically flavored link types allow the SME to specify the relationships and interdependencies among CEs, allowing the conceptual structure of the domain to be more readily grasped, similar to semantic nets. The current options are (a) IS-A (a collie IS-A dog), (b) IS-NOT-A (a dolphin IS-NOT-A fish), (c) PART-OF (a beak is PART-OF a bird), (d) CAUSES (hunger in animals CAUSES search for food), and (e) MAY-CAUSE (predatory behavior MAY-CAUSE defensive postures by the prey). On the other hand, more procedurally flavored link types allow the SME to specify the relationships among procedural steps and substeps, similar to a production system representation. This is crucial for a full understanding of procedural knowledge that can be applied to novel situations. Procedural links include the following: (a) AND (two or more nodes related by this link must co-occur), (b) OR (the condition of "A or B but not both"), (c) OR/AND (the condition of "A or B or both"), and (d) NOT (the step cannot occur). Additional link types can specify whether steps are to be performed (e) in SERIAL order (either FIXED, where steps must be accomplished in a prescribed sequence, or ANY order, where steps may be performed in any serial order) and (f) PARALLEL order (steps that are accomplished simultaneously). Finally, in addition to the available semantic and procedural links, a user-definable link allows the SME to type in a label for a relationship not already defined.

The second link option is *directionality*. This refers to the flow of control or causation between CEs. Three options exist: unidirectional, bidirectional, and no direction. These relationships are established with arrowheads attached to the end of a line. An example of the unidirectional relation, coupled with the link type CAUSES, is *Positive reinforcement—> the frequency of some behavior to increase*. A bidirectional relationship describes a balanced, symbiotic relationship between two or more elements, such as the checks and balances within the judicial, legislative, and executive branches of the U.S. government. The no-directional option, denoted by a line without arrowheads, is used between two or more nodes that do not have a clear syntactic relationship (e.g., the link between a multimedia file and its associated step in a procedure).

The third link option defines the *strength* of association. The three values for this trait—weak, moderate, and strong—indicate the degree to which the items are related. The information on strength is accomplished by varying the width of the link line (fine, medium, and bold). To illustrate, the link between fruit and kiwi would be fine, and the link between fruit and apple would be bold (at least, within North American cultures).

In general, this hybrid graphical representation of the knowledge structure is intended to make relationships among curriculum elements salient, which can thereby serve to highlight missing knowledge components. In addition, the graphical structure specifies the hierarchical dependencies between elements that are necessary for SMART to function optimally. The Network module is similar to traditional conceptual graph analysis except that with DNA, experts generate the conceptual graphs instead of the instructional designers. It is also similar to GOMS analysis in that goals, methods, and operators are represented in DNA as procedures, steps, and their logical, functional connectors.

We speculate that DNA will enable experts readily to spot and correct inadequacies (e.g., omissions, errors) in their externalized knowledge structures. Then they have the option of rectifying problems directly in the Network module or returning to the Decompose module to update the CE record. Given the shared database underlying both modules, information is easily communicated between the two. In addition, if one SME prefers to externalize knowledge and skills graphically while another prefers a text-based decomposition, both are accommodated with DNA's design.

After SMEs complete the Network module, data are stored on a floppy diskette and returned to the instructional designer. The ID reviews the CE records and conceptual graphs for glaring omissions or ambiguities in content. If there are any problems, the ID can ask the expert to expand the inadequate CEs.

Assess. The final module under development, will be used to validate the CE records and conceptual graphs generated by SMEs. This validation of externalized knowledge structures will be accomplished by having experts in the same domain review each other's data and graphs. That is, the conceptual graph and CE database created by one SME will be distributed to other SMEs, who will be requested to review and edit the information. Distribution may occur serially or in parallel. For serial distribution, the instructional designer will send out the DNA program to SME-1, who will complete it and return the output to the ID. The ID subsequently will send SME-1's output to SME-2, who will evaluate the content and structure of the initial output and return the (potentially modified) data to the ID. Depending on the degree of similarity, the ID could then send SME-2's output back to SME-1 or to a third SME. For parallel distribution, the ID will send the DNA program to multiple SMEs at the same time. Upon receipt of all their outputs, the ID will need to aggregate their data into a single knowledge structure or expert model (which may form the basis for some curriculum). This continues until the ID is satisfied that the final curriculum contains the appropriate amount of SK, PS, and CK for training needs.

Walk-Through of the Program

To make the program more concrete, we present a demonstrative walk-through of DNA, specifically the Decompose module, accomplished by one of the authors of this chapter (hereafter referred to as E1). The area chosen for illustrative purposes is knowledge and skills in using Microsoft Exchange (version. 4.0). The demonstration is based on the results from local experts' interactions with DNA decomposing this domain.

E1 began by reading the letter generated by the Customize module (see Fig. 11.1). This informed her that the ultimate goal was to produce a training system that teaches others how to create, address, and send an e-mail message using Microsoft Exchange, and to focus on providing mostly procedural elements. She was allowed to move freely between the DNA and Microsoft Exchange programs in order to execute the procedures she attempted to describe, thus verifying her description and refreshing her memory of the domain.

DNA started by presenting E1 with three procedural queries, generated from the Customize module and presented via the Main Question Queue, shown in Fig. 11.2. The expert always returns to this Main Question Queue window upon completing a particular path (symbolic, procedural, or conceptual) of decomposition.

E1 elected to respond to the first question related to creating a new e-mail message. Given that a procedural question was chosen, DNA invoked the Step Editor window (see Fig. 11.3). A variety of options exist

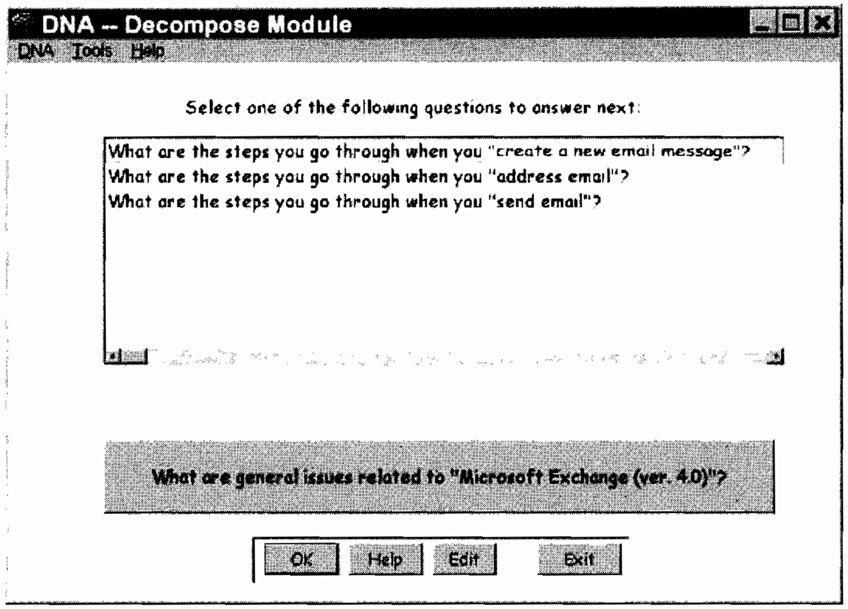

FIG. 11.2.

on the right side of the Step Editor screen for the SME to detail aspects of a procedure. Basic options are being able to add, delete, and edit steps, as well as re-order them by moving steps up and down in the Step Editor's list. Placement within the list indicates the order of execution of steps. Furthermore, because procedures can themselves be arranged hierarchically, the SME can turn a higher-level "step" into a procedure itself, with associated sub-steps, by selecting (clicking on) Sub-procedure. The Group and Ungroup options (for logically nesting steps together within a procedure) are apparent in the figure but were not operational at the time of E1's session.

Fig. 11.3 shows E1's decomposition of the procedure for creating a new e-mail message. Basically, this consists of two main steps: (a) open a new e-mail window (which can be achieved numerous ways), and then (b) compose the actual message. E1 delineated the procedure using some of the options, as well as some of the available logical operators (OR, if-then rules). For example, she outlined the multiple ways of opening a new e-mail window using if-then conditional statements, indicating three alternatives means of accomplishing the step. However, E1 described the process of composing the actual message by rendering it a sub-procedure. This is indicated by the "{more}" tag next to the last step, Compose message.

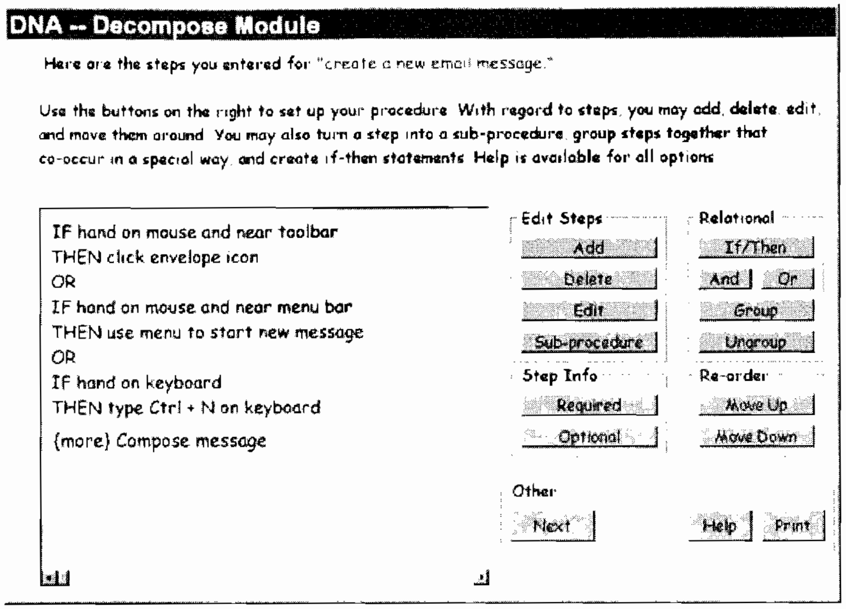

FIG. 11.3.

Figure 11.4 illustrates the interface and body of the "compose message" sub-procedure. Here, E1 indicated two steps of the sub-procedure: (a) get the cursor into the correct area (by clicking or tabbing), and then (b) compose the message (by typing, cutting and /pasting, or attaching a file). The Decompose module currently allows the expert to develop sub-procedures down two levels (to the sub-sub-procedure level).

The next two screens (see Figs. 11.5 and 11.6) illustrate the interface for creating if-then rules. This conditional was spawned from the sub-procedure "compose message" and evidences how E1 summarized two ways to get the cursor into the message area. Figure 11.5 represents the "if" part of the rule. The right-side of the screen shows one of DNA's help features—providing an example of a valid if-then input related to another domain (photography). Additional examples from other domains can be viewed by clicking the Examples button. Figure 11.6 represents the "then" part of the rule, along with another of DNA's help features (on the right). That is, pointers provide the expert with guidance on the correct way to specify input to DNA. Similar to examples, pointers are context dependent; that is, they are relevant to the current task.

After summarizing all steps and sub-steps associated with creating a new e-mail message to her satisfaction, E1 clicked the "Done" button. She

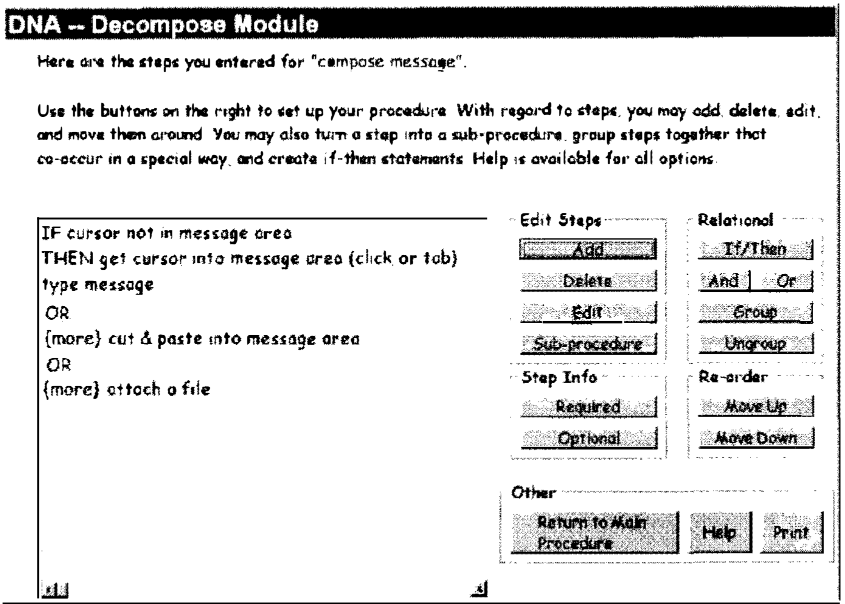

FIG. 11.4.

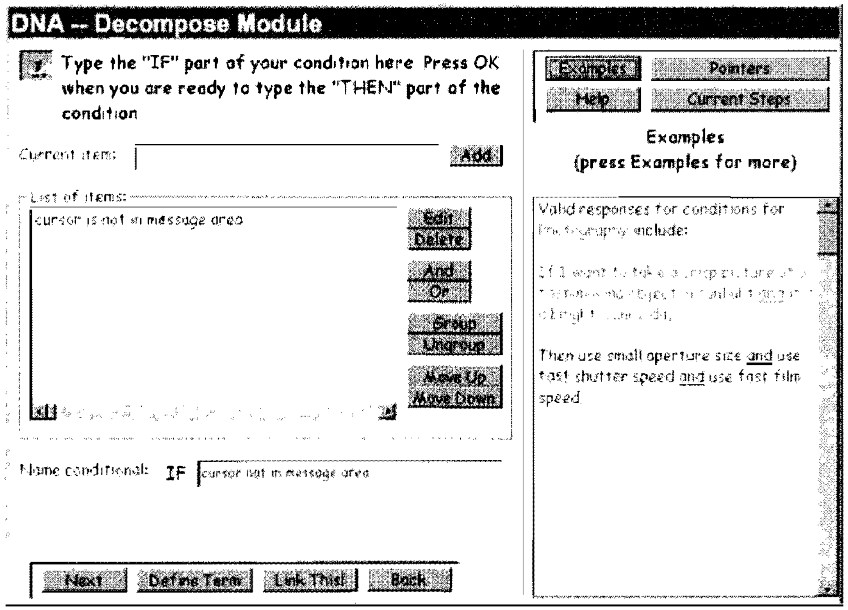

FIG. 11.5.

was then returned to the Main Question Queue, where she had the option to answer the remaining procedural queries, describe an alternative way of executing the procedure she had just delineated, or define some other aspect of Microsoft Exchange. She selected the third option (see Fig. 11.2), embodied by the large rectangular button at the bottom of the screen, "What are general issues related to Microsoft Exchange?" This choice invoked a new screen, shown in Fig. 11.7, that asks for additional symbolic, procedural, and conceptual knowledge using the "I know how to," "I can identify," and "I understand" template structures. E1 chose to describe her understanding of the importance of creating a good subject header, illustrated in Fig. 11.7. In contrast to the procedural questions she had previously encountered, this selection generated a new path of conceptually-based questions (CK).

The first CK screen that appeared asked about the components involved with choosing a good subject header. As shown in Fig. 11.8, E1 replied with two elements: relevance and brevity. The example on the right side of Fig. 11.8 shows the elements that are important to the area of photography conceptualizing how a picture gets onto some film.

After clicking on Next, the screen displayed in Fig. 11.9 appeared. E1's list of important items is shown, along with a question that requires her

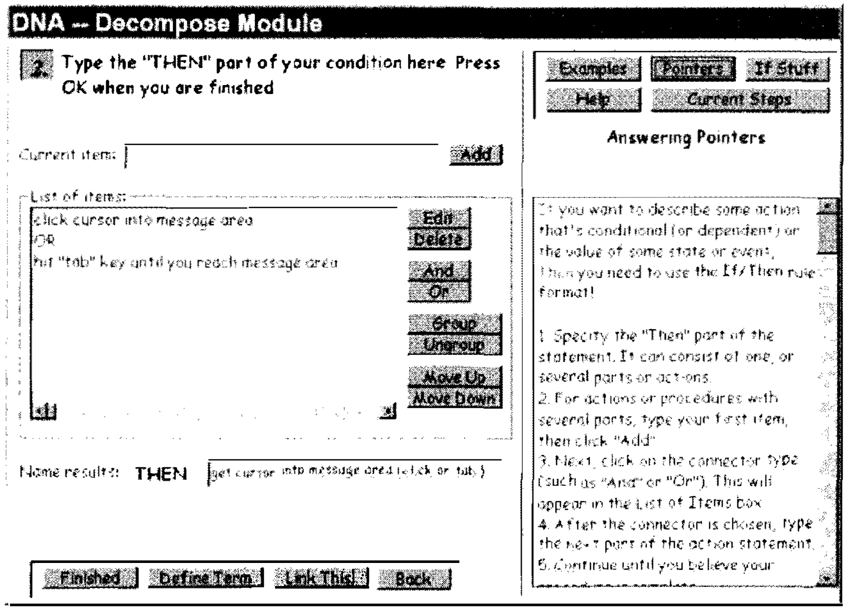

FIG. 11.6.

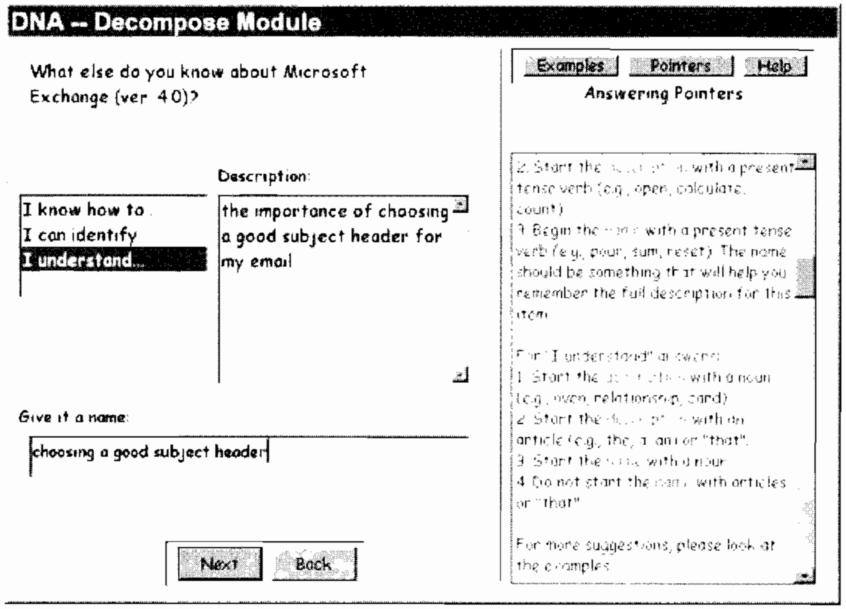

FIG. 11.7.

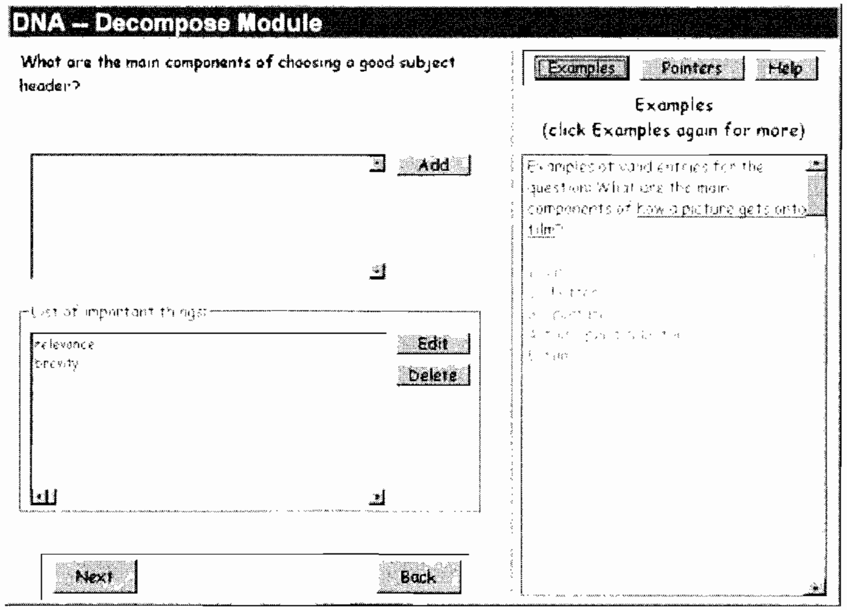

FIG. 11.8.

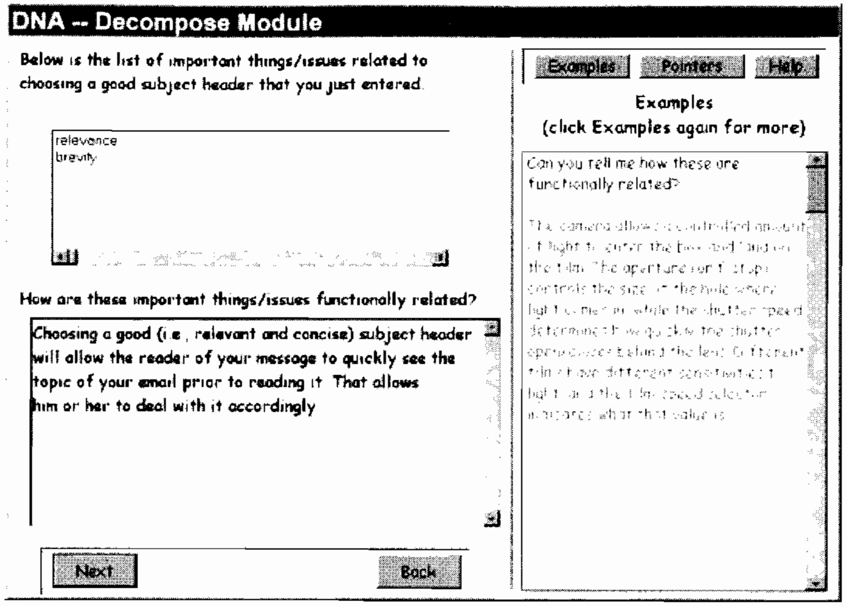

FIG. 11.9.

conceptually to "glue" the elements together in order to elaborate the current concept. The example provided to the right of the screen gives the expert an idea of what DNA seeks in the way of a response. Figure 11.9 shows how E1 wove her elements into a coherent concept. Clicking on Next took E1 to the follow-on screen (shown in Fig. 11.10), where she was asked to relate the current concept (understanding the importance of a good subject header) to the primary domain being decomposed (using Microsoft Exchange). E1's summary of the requested relationships is shown.

The last conceptually related screen (not shown) asks the expert to provide typical and atypical examples of when the current concept is useful or applicable. The expert's responses to this query can provide more information to be used by the instructional designer for developing instruction or training.

The process of answering what, how, and why questions was iterated until E1 was satisfied that all applicable questions had been answered and each higher-level goal had been decomposed into subordinate primitives. Once E1 determined she had finished decomposing her knowledge of Microsoft Exchange, she exited the Decompose module and then completed the Network tutorial before using the Network module, where she hierarchically arranged and linked the graphical nodes representing the information that she had decomposed.

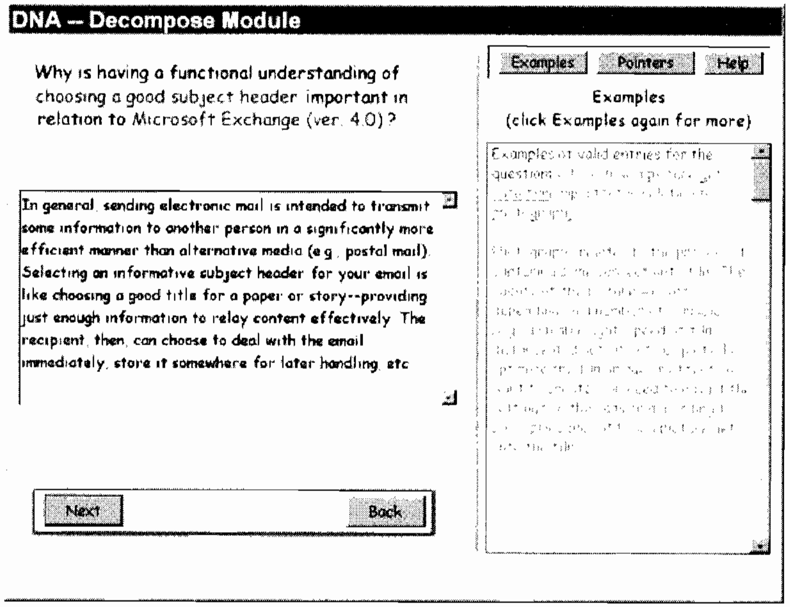

FIG. 11.10.

EI used the Network module until she was satisfied with the labeling of the links, as well as the hierarchical and spatial arrangement of the nodes representing the CEs she had delineated within the Decompose module. This structure (in theory) would then be returned to the ID to determine whether the information elicited from the expert satisfied curriculum needs.

CURRENT DESIGN AND RESEARCH DIRECTIONS

Based on the results and feedback from a handful of experts who have interacted with DNA in formative evaluations,[2] we are changing the Decompose module's interface by redesigning the content and flow of questions and developing a way to allow experts to group steps and ideas logically together to disambiguate potentially confusing relationships.

[2]So far, the program has been tested out in the areas of solving linear equations, using Microsoft Exchange, waiting tables in a restaurant, and measures of central tendency, all fairly constrained and mostly procedural domains.

Flow of Questions

Originally the Decompose module asked a series of questions to elicit the rationale of each step of a procedure as it had been delineated. However, several experts noted that their train of thought was disrupted during procedural decomposition. Consequently, we now present these queries in the form of local follow-up questions after the SME has completed the delineation of a procedure into all of its constituent steps. We are also developing a series of global follow-up questions that will be presented on completion of the entire Decompose module. These questions are intended to obtain more general information about the domain, such as typical problem areas and appropriate analogies.

Local Follow-Up Questions

When implemented, local follow-up questions pertaining to each procedure will require the SME to reflect on the immediate goal being served, as well as the rationale behind the structure of individual steps that have been delineated. Examples (where *X* refers to the current procedure) include questions such as, Why do you do X? and What are typical and atypical situations in which you would do X? In addition, when an exclusive disjunction (e.g., “A or B”) has been stipulated, the SME will be queried as to which factors influence whether A versus B would be done or occur. Similarly, when a conditional (e.g., “If A, then B”) is outlined as part of a procedure, the SME will be prompted to consider its logical consequences to ensure that alternative cases have not been overlooked. Some of these questions include: When A does not hold, should one still do B? and Are there other common conditions that should trigger doing B? Thus, local follow-up questions posed when an SME finishes outlining the steps of a procedure will serve to disambiguate specifications of a given procedure, as well as elaborate symbolic and conceptual knowledge related to it.

Global Follow-Up Questions

To aid the instructional designer in generating curriculum, follow-up questions will be posed to the SME on completion of the Decompose module. They will attempt to elicit the SME's overview of the field–that is, the themes and principles of the domain being decomposed. Some global follow-up question examples include (where X is the domain being decomposed): What are some difficult areas you have encountered in the acquisition of X? What has worked for you in repairing this impasse? Can you describe a connection between this field and other domain? and What are some good analogies that help in understanding some aspect of the domain? In sum, local and global follow-up questions support DNA's effort to elicit information on which the instructional designer can generate potential examples, problem scenarios, questions, analogies, and real-world applications related to the domain to be instructed.

Grouping (and Ungrouping) Elements

The next version of DNA will incorporate changes to the Step Editor to help clarify the delineation of procedures. Specifically, experts will be allowed to group related CEs, thereby providing syntactic structure in relation to the procedure. For example, steps of a procedure such as "do A and B or C" are ambiguous unless syntactically structured to clarify whether [(A AND B) OR C] or [A AND (B OR C)] is the intended representation. These differences can be crucial in identifying the proper execution of a procedure. Once operational, the Group and Ungroup options will be available for all procedural decomposition in the Step Editor interface, as well as in the subprocedure and conditional windows. The SME will be able to left-click, and thereby highlight, steps listed in the Step Editor to identify the items to be grouped together. Once they are highlighted, the SME will click the Group button. That will result in a marker's being placed next to the selection to indicate the grouping. For example, setting up the grouping of [(A AND B) OR C] would show A and B having a "1" next to them, indicating their conjoined status at the first, most nested level.

There will be up to three levels of hierarchical grouping possible, specified from inner (most nested) to outer (most general) organizations. Ungrouping will work in the opposite manner, allowing the expert to unlink elements progressively from highest to lowest level groupings.

Summary

The local and global follow-up questions are expected to yield elaborated CEs by clarifying specific procedures and their rationales, as well as by characterizing the domain as a whole. In addition, the Grouping function will support clarification of procedures. Consequently, the instructional designer should have sufficient and rich information to facilitate curriculum development.

CONCLUSIONS

We are concerned with enhancing the development of intelligent instructional systems by focusing on, and automating, the process to obtain the "what" of these systems (i.e., expert model).

Heretofore, the practicality of using traditional elicitation and organization methods has been limited due to a number of reasons. First, traditional acquisition methods are very costly in terms of time, money, and skilled personnel resources. Second, they tend to be limited in scope of applicability. Finally, the specifications for using the various methods are largely unclear; they are more art than science. DNA can potentially deal with these shortcomings.

The costs associated with developing expert models for automated instruction can be reduced. Typically this development cycle involves a number of persons who perform different tasks across various points in time. For example, knowledge engineers elicit information through interviews, observations, and other techniques. Subsequently this information is transcribed. The transcription must then be simplified or coded into units representing discrete actions and bits of knowledge. Finally, cognitive psychologists arrange the information so that it suits the specific purpose for which the analysis was conducted—for example, expert model, cognitive simulation, or design of human–computer interface. In response to this resource–cost issue, DNA condenses these stages into one symbiotic, standardized process of eliciting and organizing knowledge. The result of this collapse is that the time overall to develop an expert model should be reduced because transcription and codification tasks are no longer necessary.

The scope of applicability of many traditional knowledge elicitation and organization methods is typically narrow. That is, many methods are designed to be domain or task specific. For example, Precursor, Action, Response, Interpretation (PARI; Hall, Gott, & Pokorny, 1995) has been shown to be useful in delineating troubleshooting procedures, particularly in relation to avionics tasks. DNA has been designed to apply to both procedural and conceptual domains. This makes it a broadly applicable tool in terms of specifying curricula for a variety of topics.

Finally, specifications for conducting analyses can be clarified. Schraagen et al. (1997) reviewed the current state of cognitive task analysis techniques and concluded, "Few integrated methods exist, that little attention is being paid to the conditions under which methods are appropriate, and that often it is unclear how products of CTA should be used" (p. 5). In response to this criticism, DNA was designed to automate and thus standardize the analysis process. This standardization is intended to reduce variance in the way that the technique is employed. However, because of inherent differences in knowledge representations, actual DNA outputs may still show variance among experts. In addition, its design is based on explicit principles of learning and instruction as embodied in SMART, a validated instructional tool. DNA's purpose is to obtain domain knowledge for intelligent instructional systems. Therefore, the use of DNA's output is readily apparent.

In addition to these features, DNA was designed to be user friendly. This extends the groups of people who can use such a tool to elicit the knowledge and skills underlying a particular domain or task. This contrasts with the few existing automated CTA programs (Hamilton, 1997; Williams, 1993; Zachary, Zaklad, Hicinbothom, Ryder, & Purcell, 1993), which typically require input by either programmers or human factors personnel.

We are well aware that DNA requires extensive empirical research. Consequently, we will be assessing its effectiveness and efficiency. Specifically, we will conduct a series of investigations, starting with basic questions: Can DNA be used to obtain meaningful data that parallel existing data in the same domain, elicited by traditional means? Our initial test of this question involves several statistics experts using DNA to decompose and network measures of central tendency. This allows us to determine the degree of similarity between an existing and effective expert model (i.e., Stat Lady, DS-2 module; Shute, Gawlick, & Lefort, 1996) and DNA-obtained data for the same topic. Preliminary data from this investigation are quite encouraging (see Shute, Torreano, & Willis, 1998).

In addition, we will investigate DNA's efficiency relative to other elicitation techniques. Specifically, can DNA obtain knowledge structures comparable to those obtained by traditional knowledge elicitation methods but more quickly? Other immediate research questions will examine the capabilities and limitations of DNA in relation to the types of domains, tasks, and purposes for which it is best suited. Some basic research questions that we will explore with DNA include examining novice-to-expert transitions within disparate domains and comparing knowledge representations underlying different levels of expertise within the same domain.

DNA promises to be a useful knowledge elicitation and organization tool for developing curriculum, representing a good first step towards opening up the bottleneck in intelligent tutoring system development as related to the expert model. Using DNA in conjunction with SMART should streamline this process. That is, DNA was developed to work in concert with SMART, deriving curriculum elements from experts that fit a particular instructional framework. This provides a rich database of underlying knowledge and skill elements that can be subsequently monitored by SMART with regard to learner's acquisition or mastery status. The degree to which this bottleneck is opened, however, will not be known until data come in from these necessary evaluations.

ACKNOWLEDGMENTS

The research reported in this chapter was conducted as part of the Air Force Research Laboratory, TRAIN Project, Brooks Air Force Base, Lackland, Texas. Funds for this research were provided by the Air Force Office of Scientific Research. In addition, this work was performed while the second author held a National Research Council AFRL Research Associateship. The opinions expressed in this chapter are those of the authors and do not necessarily reflect those of the U.S. Air Force.

REFERENCES

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

Anderson, J. R. (1988). The expert module. In M. C. Polson & J. J. Richardson (Eds.), *Foundations of intelligent tutoring systems* (pp. 21–50). Hillsdale, NJ: Lawrence Erlbaum Associates.

Anderson, J. R. (1993). *Rules of the mind*. Hillsdale, NJ: Lawrence Erlbaum Associates.

Card, S., Moran, T. P., & Newell, A. (1983). *The psychology of human-computer interaction*. Hillsdale, NJ: Lawrence Erlbaum Associates.

Collins, A. M., & Quillian, M. R. (1969). Retrieval time from semantic memory. *Journal of Verbal Learning and Verbal Behavior, 8*, 240–247.

Durkin, J. (1994). Knowledge acquisition. In J. Durkin (Ed.), *Expert systems: Design and development* (pp. 518–599). New York: Macmillan.

Gagné, R., & Briggs, L. (1965). *The conditions of learning*. New York: Rinehart & Winston.

Gordon, S. E., Schmierer, K. A., & Gill, R. T. (1993). Conceptual graph analysis: Knowledge acquisition for instructional system design. *Human Factors, 35*, 459–481.

Gray, W. D., John, B. E., & Atwood, M. E. (1993). Project Ernestine: Validating a GOMS analysis for predicting and explaining real-world task performance. *In Human-computer interaction* Vol. 8, pp. 237–309.

Hall, E. P., Gott, S. P., & Pokorny, R. A. (1995). *A procedural guide to cognitive task analysis: The PARI methodology* (Rep. No. AL/HR-TR-1995-01 08). Brooks Air Force Base, TX: U.S. Air Force Material Command.

Hamilton, W. I. (1997). *Cognitive task analysis using ATLAS*. NATO-ONR Workshop on Cognitive Task Analysis, Washington, DC.

Hayes-Roth, F., Waterman, D. A., & Lenat, D.B. (1983). *Building expert systems*. Reading, MA: Addison Wesley.

Jonassen, D. H., Beissner, K., & Yacci, M. (1993). *Structural knowledge: Techniques for representing, conveying, and acquiring structural knowledge*. Hillsdale, NJ: Lawrence Erlbaum Associates.

Lajoie, S. P., & Derry, S. J. (1993). *Computers as cognitive tools*. Hillsdale, NJ: Lawrence Erlbaum Associates.

Newell, A. (1990). *Unified theories of cognition*. Cambridge, MA: Harvard University Press.

Polson, M. C., & Richardson, J. J. (1988). *Foundations of intelligent tutoring systems*. Hillsdale, NJ: Lawrence Erlbaum Associates.

Psotka, J. Massey, L. D., & Mutter, S. A. (1988). *Intelligent tutoring systems: Lessons learned*. Hillsdale, NJ: Lawrence Erlbaum Associates.

Ryder, J. M., & Redding, R. E. (1993). Integrating cognitive task analysis into instructional systems development. *Educational Technology Research and Development, 41*, 75–96.

Schraagen, J. M. C., Chipman, S. E., Shute, V., Annett, J., Strub, M., Sheppard, C., Ruisseau, J.-Y., & Graff, N. (1997). *State-of-the-art review of cognitive task analysis techniques* (Report of RSG.27 on Cognitive Task Analysis NATO Defence Research Group, Panel 8/RSG.27). (The Netherlands) TNO Human Factors Research Institute Group: Information Processing.

Shute, V. J. (1994). Learning processes and learning outcomes. In T. Husen & T. N. Postlethwaite (Eds.), *International encyclopedia of education* (2nd ed., pp. 3315–3325). New York: Pergamon Press.

Shute, V. J. (1995). SMART: Student modeling approach for responsive tutoring. *User Modeling and User-Adapted Interaction, 5*, 1–44.

Shute, V. J., & Catrambone, R. (1996, July). Unified vs. tailored analogies: Effects on conceptual knowledge acquisition. In *Proceedings of the ICLS 96 Conference*. Washington, DC: Association for the Advancement of Computers in Education.

Shute, V. J., Gawlick, L. A., & Lefort, N. K. (1996). *Stat Lady: Descriptive Statistics Module 2* [Unpublished computer program]. Brooks Air Force Base, TX: Armstrong Laboratory.

Shute, V. J., & Psotka, J. (1996). Intelligent tutoring systems: Past, present, and future. In D. Jonassen (Ed.), *Handbook of research on educational communications and technology* (pp. 57–600). New York: Macmillan.

Shute, V. J., Torreano, L. A., & Willis, R. E. (1998). DNA–Uncorking the bottleneck in knowledge elicitation and organization. In *Proceedings of the 4th International ITS 98 conference*, San Antonio, TX.

Shute, V. J., Willis, R. E., & Torreano, L. A. (1998). *DNA—Version 2.0* (Automated Cognitive Task Analysis Tool). [Unpublished computer program/prototype]. Brooks Air Force Base, TX: Armstrong Laboratory.

Sleeman, D. H., & Brown, J. S. (1982). *Intelligent tutoring systems*. London: Academic.

Williams, K. E. (1993). *The development of an automated cognitive task analysis and modeling process for intelligent tutoring system development*. Contract final report on N00014-91-J-5-1500. Manpower Personnel and Training Program, Office of Naval Research.

Zachary, W. W., Zaklad, A. L., Hicinbothom, J. H., Ryder, J. M., & Purcell, J. A. (1993). COGNET representation of tactical decision-making in anti-air warfare. In *Proceedings of the Human Factors and Ergonomic Society 37th Annual Meeting* (pp. 1112–1116). Seattle, WA. Human Factors & Ergonomic Society.