Technology Transfer Using Automated Knowledge-Acquisition Tools

Gerald L. Atkinson

In the past, the clear dominance of U.S. strategic nuclear forces permitted the deployment of tactical nuclear and conventional forces at a relatively high but acceptable level of risk with respect to survivability. Today, however, the U.S. strategic nuclear posture no longer provides the same dominance. Should deterrence fail, the United States must be prepared to engage in combat across the full spectrum of possible conflicts, including conventional operations in a nuclear environment. Thus, early in the 1980s, the Department of Defense (DOD) (1983) issued a directive that required weapon systems be able to survive the effects of nuclear weapons. This directive attempted to assure that meaningful survivability programs would become integral parts of the total defense weapon acquisition system. For the first time in U.S. history, this effort explicitly included the mandate that funding for nuclear survivability and hardening be included in investment planning for U.S. forces. (Although in some cases, techniques other than hardening to nuclear effects might be feasible, nuclear hardening continues to be the centerpiece for the survivability of weapon systems. The phrase nuclear hardening refers to the art of designing weapon systems and their

components so that they are less susceptible to the effects of nuclear weapon detonation than if left unprotected.)

Many of the program managers, technical directors, scientists, and engineers within the U.S. government agencies responsible for carrying out the directive recognized that it would be difficult to implement. This realization stemmed from the fact that nuclear hardening of weapon systems is not an established engineering discipline, and much remains to be known about nuclear weapon effects environments, simulation testing, and system hardness validation. The Defense Nuclear Agency (DNA) responded to this challenge with several innovative practices. One was the support of technology transfer activities that showed promise for expanding the nuclear hardening expertise to military program offices through the development of expert systems that help users evaluate the hardness of their weapon system to nuclear effects. The development of these expert systems and the development of the automated knowledge-acquisition tool Knack, which is used to generate the expert systems, is the topic of this chapter. It is through these software tools that technology transfer in the domain of nuclear effects hardening is being extended to the larger number of program managers and their staff members who must be involved.

Technology Transfer Using Expert Systems

It was well understood by nuclear hardening experts that the requirement placed on the military services to assure the survivability of their weapon systems during and after a nuclear engagement placed a severe strain on the nations hardening design resources. There simply was not a sufficient supply of trained and knowledgeable experts available nationwide to assist in the hardness design of these weapon systems. The United States Army with over 140 weapon systems within the acquisition phase, the United States Navy with over 200 shipboard systems to be hardened, and the United States Air Force with a similar number of systems that must be certified survivable places an overwhelming demand on the limited resources of available talent. It was clear that a means must be found to multiply the available expertise in hardness design techniques and practices through some technology that assisted in technology transfer. DNA resolved to pursue the development of efficient computer-based methods of technology transfer.

After consulting with members of the Defense Advanced Research Projects Agency, DNA decided to enlist the aid of John McDermott at Carnegie-Mellon University (CMU) to lead a development program that would eventually result in a set of expert systems and an automat-

ed knowledge-acquisition kernel that would generate them. This group, along with several private contractors, produced the Knack knowledge-acquisition tool as well as six expert systems through which the nuclear hardening technology transfer takes place (Atkinson 1988). These software tools are being used by the Harry Diamond Laboratories (HDL) in everyday practice. They are showing productivity improvements by factors of 10 to 30 in carrying out reporting activities required during the acquisition cycle of Army systems. The technology transfer of weapon system hardening design practices by these expert systems has been reported in a DNA journal (Atkinson 1988) and at a symposium (Atkinson 1989).

Knowledge Acquisition

The first generation of expert systems was constructed by an initial process of discourse between a knowledge engineer and a human expert, where the knowledge engineer extracted domain-specific knowledge, adding it to a growing knowledge base usable by a computer program—the expert system. The expert must communicate not merely the facts of his(her) field but also the *heuristics*, the informal judgmental rules that guide him(her). These rules are rarely thought about concretely and almost never appear in journal articles, textbooks, or university courses. Such heuristics are difficult to verbalize. According to Lenat (1983), "The critical stage of this process, the limiting step, is the transfer of expertise. From the program's point of view, the limitation is the slow rate at which it acquires knowledge. This is the central problem facing knowledge engineering today, the bottleneck of knowledge acquisition."

During the 1980s, knowledge acquisition continued to be the major stumbling block to the efficient development of expert systems within reasonable time and cost constraints. Even with the introduction of expert system generators or shells, the knowledge acquired during interviews with experts was still the basis for the expert system, and the process of acquiring this knowledge was and still is fraught with difficulty. The technical gulf between the domain expert and the knowledge engineer is often unbridgeable. The concepts, ideas, language, and backgrounds for each can be foreign to the other. Many times, the expert is simply unable to articulate how he/she arrives at a solution to a task or problem, at least in a form that can easily be captured by the knowledge engineer. The knowledge-acquisition bottleneck to building expert systems has been addressed by many researchers but has not been eliminated (Boose 1988). The CMU researchers found a measure of

success by narrowly focusing the application domain. This technique describes the path taken by the developers of the automated knowledge-acquisition kernel Knack.

Automated Knowledge Acquisition

University research has established the theoretical basis for automating the knowledge-acquisition process for certain classes of applications (Chandrasekaran 1983). The basic idea is that a large knowledge base can be kept maintainable by organizing it according to the different roles that knowledge plays. *Knowledge roles*, the organizational units of the knowledge base, are made explicit by defining a problem-solving method. Without such an organizational structure, the huge number of knowledge-acquisition tools and techniques that have been devised, some for relatively general problems, break down when used for serious applications. Consequently, most serious expert system developers can use shells or expert system generators that are on the commercial market today for prototype development but resort to custom-built inference engines and knowledge bases when constructing the actual application (Leaman 1989). In addition, industry is finding that it is much more practical to provide AI training to engineers experienced in the technical knowledge domain rather than use knowledge engineers to ferret out the knowledge from domain experts and encode it. All this experience is evidence of the need for automated special- purpose knowledge-acquisition tools (that is, expert system generators) that can be used by a technical domain expert to build an expert system without a knowledge engineer or AI practitioner (Rowan 1989).

Recent research has implemented the theory of knowledge roles by constructing specialized inference engines suitable for a narrowly defined knowledge domain and automating the acquisition of knowledge within this domain by using a specialized knowledge-acquisition kernel (Marcus 1988). This kernel is a software program that generates computer code encapsulating the domain knowledge. This domain knowledge can then be fed to an inference engine that processes the knowledge base as it works toward a solution to the problem at hand. This research has defined a range of role-limiting methods that were implemented by CMU workers. The idea behind this research is that if the AI discipline is serious about separating the knowledge base from the control mechanism (inference engine), then it should be possible to devise a set of role-limiting methods (inference engines), where each method defines the roles that the task-specific knowledge must play and the forms in which this knowledge can be represented.

A simple role-limiting method typically consists of a simple loop over a sequence of 5 or 10 steps. Some of the steps operate on significant bodies of task-specific knowledge. Within a step, there is no control; that is, it makes no difference in what order the actions are performed. It is the regularity of the task-specific knowledge that makes automating the building of expert systems a tractable task. Knack incorporates two role-limiting methods: The acquire-and-present method produces a report, and the propose-and-revise method contributes to the evaluation of the hardness of an electromechanical system to nuclear effects (Klinker et al. 1987). The Knack automated knowledge-acquisition kernel is the central element in the generation of six expert systems by which the technology transfer of weapon system hardness design is carried out. The use of the Knack software is described in the remainder of this chapter.

The Nuclear Hardening Design Problem

The problem associated with assuring the hardness of weapon systems to the effects of nuclear weapons is illustrated in figure 1. The energy resulting from a nuclear burst can couple to an electromechanical system through the system's response to the electromagnetic pulse (EMP), initial nuclear radiation (INR), and blast and thermal effects. Voltages resulting from the electric field associated with the burst can couple to the outside cable and be transferred to sensitive internal equipments through internal cables unless protected by the hardening features of the design. Energy can diffuse through apertures such as windows and cable entry panels and set up internal electric fields around sensitive equipment.

The expert system required for the nuclear hardening design problem must have two features. It must evaluate the hardness of the system components to nuclear weapon effects, and it must discuss this evaluation, along with other required information, in a report that is consistent with the requirements of the weapon system acquisition phase. For example, the evaluation of the hardness of system components that must be protected against EMP must discuss voltage, current, inductance, and other parameters of the pathway from the outside cable to the equipment interface circuits within the enclosure. The expert system must ascertain the system and component properties by interacting with the user (a design engineer, program manager, or hardness evaluator) and must calculate whether the system, as presented, is hard to the nuclear weapon effect. This function of the expert system is analytic.

If the evaluation is that the component or system is not hard to the effect, the expert system must recommend a fix or engineering solu-

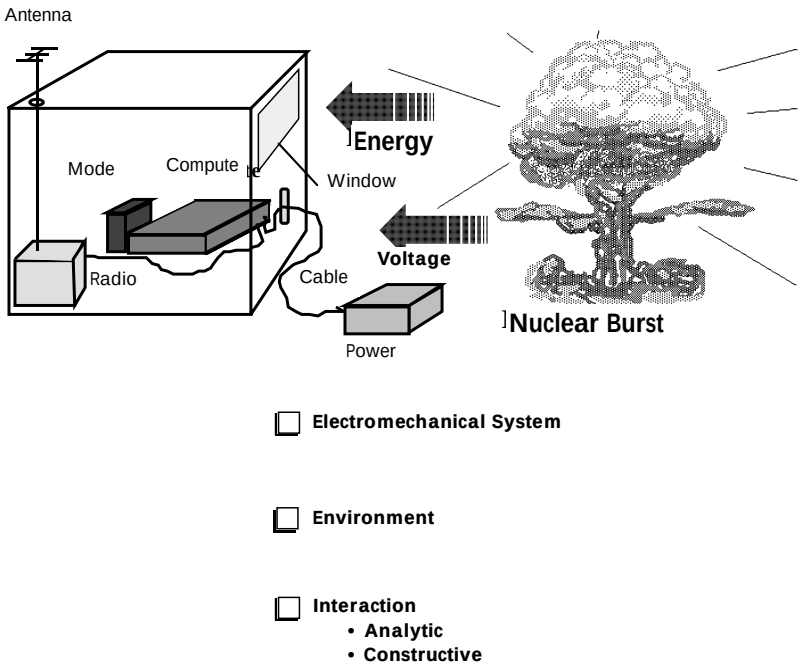


Figure 1. The System Hardening Problem.

tion, which is the constructive function of the expert system. For example, it might first recommend that a shield be designed for the outside cable. If the expert system then evaluated the system would remain inadequately hardened after this fix, it might recommend that a terminal protection device be included in the design of the cable-to-enclosure interface. This interaction continues until the component or system is eventually evaluated as hard to the nuclear weapon effect; if it is not hard, other means outside the expertise of the expert system must be invoked to address the problem. The results are contained in a report generated by the expert system after user interaction is completed.

User interaction with the expert system is dictated by a domain expert who must build the expert system using Knack. For example, the domain expert must construct the calculational pathways that the expert system uses to make a hard–not hard determination. In addition, the domain expert suggests the fixes and their priority for assignment by the expert system. The most important feature of this process is that the domain expert accomplishes these tasks without having to know any programming language, any AI representation schemes, anything

about the semantics of rules, or any other computer science abstractions. The Knack tool is a state-of-the-art specialized expert system generator that allows interaction with the domain expert through windows, menus, graphic displays, and prompting devices. These devices allow the expert to build the expert system knowledge base (encoded in OPS5 rules) and use his(her) own engineering subset of the English language. The expert graphically builds a semantic net, the objects of which contain the knowledge base, without being aware that he/she is building such a net. The expert selects strategies by which the expert system draws the required information from the expert system user. The expert acquires this information by interacting with Knack, selecting among alternatives suggested by the Knack heuristics. A discussion of this interaction between the domain expert and Knack constitutes the remainder of this chapter.

Knack Automatically Generates Knowledge Base Code

The technical domain expert interacts with Knack to generate knowledge base code. This code is in the form of OPS5 rules. The process by which this code is generated is depicted in figure 2. The output of a Knack session is a set of 10 *working memory element* (WME) files that contain strings that can be translated into OPS5 WMEs. These files contain the knowledge base as it exists at any point in time. When the domain expert is satisfied that he/she has properly designed the knowledge base, three of the WME files are input to the Knot translator. This software translates the appropriate strings into OPS5 rules that are stored in the three OPS files (skeleton.ops, strategy.ops, and report.ops). These files are the knowledge base that is then compiled with the expert system's inference engine (the Wringer [writes report from information gathered by evaluation and review] kernel is a set of 15 files containing OPS5 rules) to produce the Wringer expert system.

The most important feature of this process is the automatic generation of OPS5 code; no programming is required by the technical domain expert. This feature eliminates the knowledge engineer from the process, diminishing the knowledge-acquisition bottleneck. Diminishing this bottleneck is possible only because of the narrow technical domain of the application, that is, writing an evaluation.

Automatic Generation of OPS5 Code

Building an expert system during a Knack session involves four steps: (1) building a domain model by manipulating graphic objects (rectan-

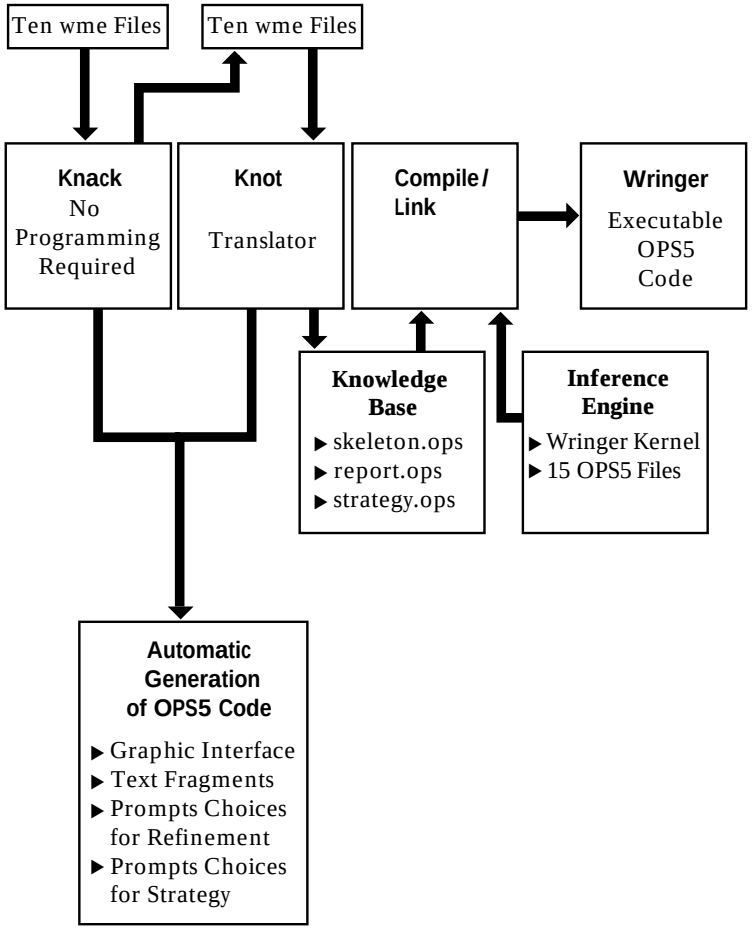


Figure 2. The Generation of Knowledge Base Code by Knack.

gles, parallelograms, and ellipses) and filling in a table containing characteristics and their values for each graphic object (this simple process allows Knack to construct a semantic net of the knowledge domain objects, their characteristic values, and their relationships, (2) typing a sample report that characterizes all the hardness design information required for a range of weapon systems at a particular stage in their acquisition, (3) interacting with Knack prompts to select strategies by which the expert system can obtain information from a user, and (4) choosing constraints (comparison of threat and safe values of a characteristic) that must be met during an evaluation of hardness as well as choosing fixes and their priorities for overcoming violated constraints. No knowl-

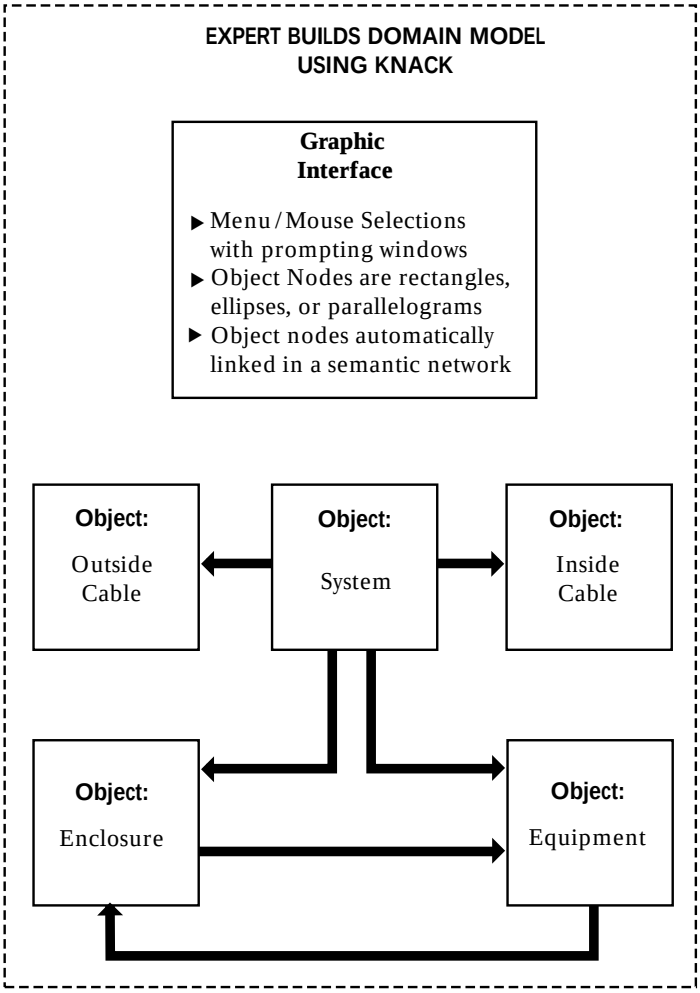


Figure 3. Building a Domain Model Using Knack.

edge of, or experience with, any programming language or AI representation constructs is necessary; the domain expert uses his(her) own vocabulary and concepts in designing the expert system.

Building the Domain Model

The domain expert selects "domain model" from a main menu, and Knack presents a window within which graphic objects can be con-

structed using a simple mouse-clicking operation. Figure 3 is a simple representation of the kinds of objects and their links that result from this operation.

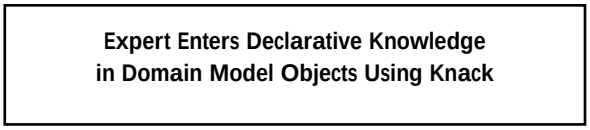
By selecting commands from a menu, the domain expert can add, delete, move, or rename objects in the domain model. He/She establishes the relationship among objects by answering queries from Knack about the object's parent as well as organizing the links among objects joined by a connection component, such as an outside or inside cable (figure 3). He/She constructs this domain model as a top-down hierarchy of his(her) system design, with system components, subcomponents, and so on. In addition, he/she constructs the hierarchy of environment effects (EMP, INR, blast, and thermal) that he/she knows about and describes them to a detail level that is consistent with his(her) proposed evaluation of the interaction between the environment and the weapon system. In this manner, the domain expert is building a semantic net of relationships among objects that interact in ways that he/she formulates later.

Each node in the domain model can have characteristics defined for it by the domain expert. He/she can assign example values for each of these characteristics. Knack uses these values to variablize the sample report fragments and produce an expert system that reports on a range of weapon systems of different types. (The term variablize means to replace a constant key word with a variable name, a place saver, which can take on different values during different Wringer interactions with a user.) Figure 4 portrays the table presentation that Knack provides to the domain expert. The domain expert defines the characteristics of his(her) generic system and nuclear weapon effect environment. This table of values for each of the nodes in the domain model establishes the parameters and some example values of the domain; this information, along with the relationships established by the links in the domain model, make up the knowledge base for the expert system under construction.

When the domain expert has completed constructing the domain model, he/she is ready to input the text of a sample straw man report into Knack. This sample report is the model for the expert system's report and is an important document. It must be flexible enough to capture the variability that is needed such that the expert system's report applies to at least 90 percent of the applicable weapon systems.

Typing in the Sample Report

The domain expert interacts with Knack to type in the sample report in fragments. These fragments can be sentences or paragraphs. Figure



Declarative Knowledge Entered in Graphic Table

- Objects have characteristics (attributes)
- Characteristics have example values

Characteristic	Example Value	Constraint
Name	computer, radio	
Safe Voltage	120	
Threat Voltage	number	<=< equipment. safe-voltage>
Weight	10	

Figure 4. Table Used to Enter Declarative Knowledge.

5 displays an example of a report fragment (the topmost example).

The construction of these fragments allows the report text variability that is incorporated in the expert system. That is, the domain expert must carefully select example values within the text of the fragments that precisely match the example values for object characteristics he/she defined in the domain model. It is this precise matching that allows Knack to automatically variablize the report fragment text. This variablization produces report text that then applies to at least 80 percent of all possible weapon systems that are in the acquisition cycle. Consequently, the domain expert must be an expert in the reporting functions as well as the range of weapon systems for which the expert system is to be used. After the report fragments are input into Knack, the domain expert interacts with Knack to variablize them.

Variablizing the Report Fragments

After typing in the sample report, the next step is to integrate the report fragments with the domain model. This integration is accomplished by a parser-matcher within Knack that variablizes the example values in the report fragments that match the example values in the domain model. Figure 5 displays an example of a variablized report fragment (the second example in the figure). Here, Knack recognizes

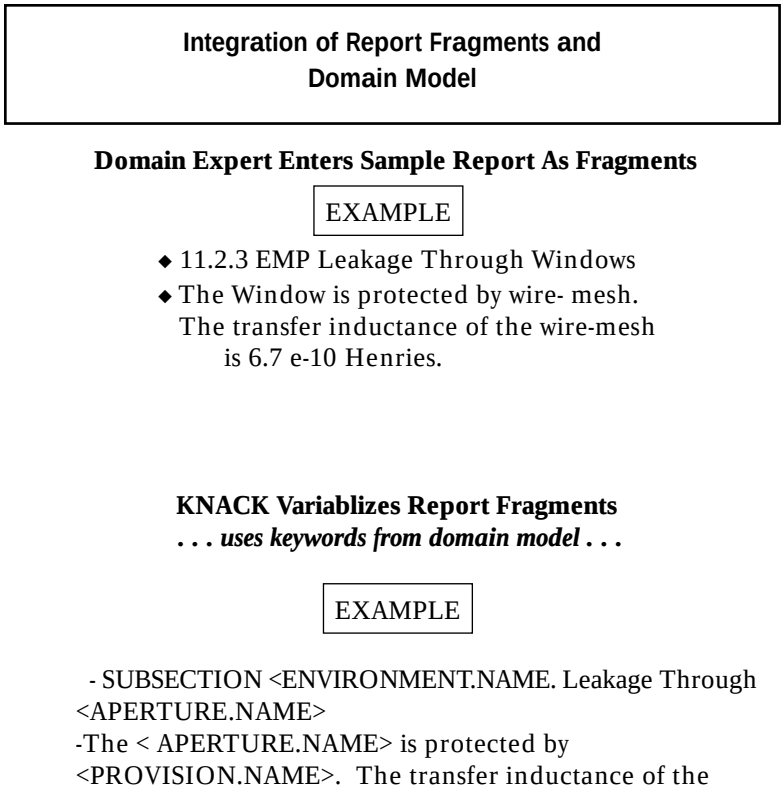


Figure 5. A Report Fragment.

that the digit string, 11.1.3, is a subsection identifier in some formatting language and substitutes the appropriate formatting language command for a subsection in the variablized text. The example value, EMP, appears in the fragment and in the domain model as the name of a nuclear weapon effect environment. The example value, Windows, appears in the report fragment, as well as in the domain model, for the name characteristic of the object called aperture. Knack carries out the appropriate variablization by matching these sample values and constructing a variablized text. Similarly, Knack recognizes wire-mesh as the example value in the domain model for the name characteristic of

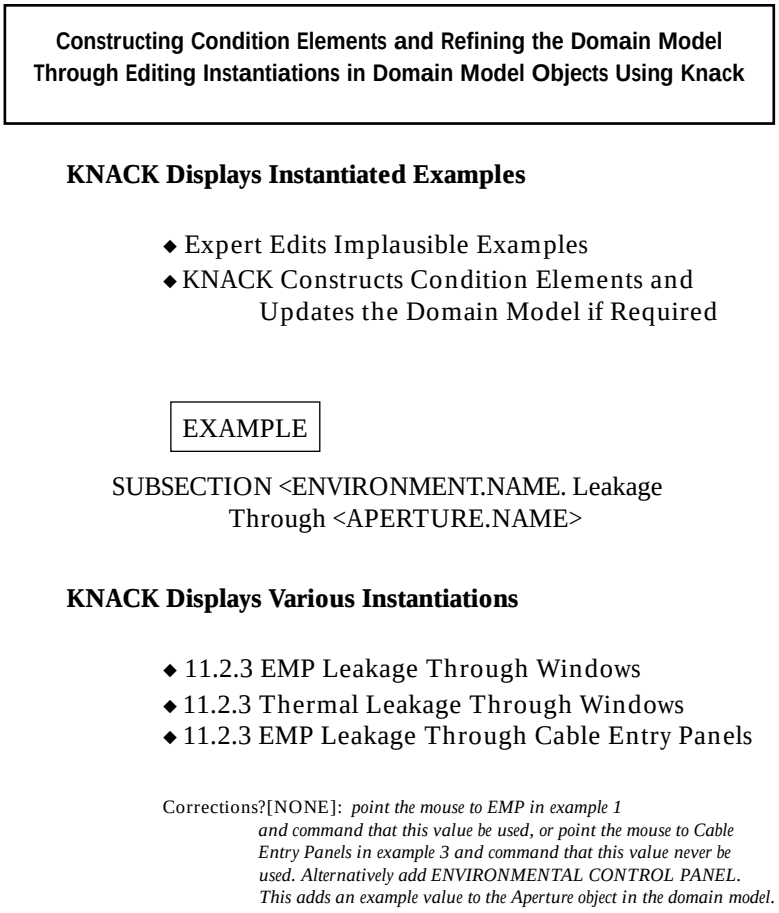


Figure 6. Constructing Condition Elements and Refining the Domain Model.

the provision object in the domain model and carries out its variablization in the text. Finally, Knack recognizes the floating-point number 6.7 e-10 as the example value in the domain model for the transfer inductance characteristic of the object provision. In this way, Knack automatically variablizes the text fragments that were input into Knack by the domain expert.

Refining the Variablized Text

A further step in the variablization process occurs when Knack instantiates the variablized report fragment with all possible example values

that the expert entered in the domain model for each of the variables (that is, object-characteristic pairs). Knack uses heuristics that only display three possible instantiations of the fragment at one time out of all possible combinations of example values for all the variables. This heuristic is based on the assumption that any human expert can focus on, at most, three possible choices in a complex problem domain. The expert edits these Knack-generated instantiations by interacting with Knack through menus and prompts for clarifying information. Figure 6 provides an example of the process by which Knack suggests possible instantiations of the variablized text fragments; the domain expert edits these instantiations through Knack menus or a simple text editor that appears in a separate window. This interaction is accomplished using the vocabulary and text that the domain expert used during the construction of the domain model and the sample report fragments. In fact, Knack uses this interaction to construct strings that eventually end up as OPS5 rules with condition elements from the expert's editing of the Knack-suggested instantiations.

In the example in figure 6, the domain expert interacts with Knack to make one of several possible selections from the menus provided. In one instance, he/she can demand that the value EMP be the only value used for the text fragment. Thus, of the complete set of EMP, INR, blast, and thermal nuclear weapon effects, the expert specifies that only this fragment makes sense for EMP. No other effect leaks through windows. This selection places a condition element in the left-hand side of the rule that produces this report fragment for the expert system when fired. In addition, the domain expert can specify that the value "cable entry panel" can never be acceptable for the aperture object. That is, he/she might know some technical reason why EMP never leaks through cable entry panels. This choice produces further constraints in the condition elements on the left-hand side of the appropriate OPS5 rule constructed by Knack without the domain expert being aware of such a rule-constructing process.

Finally, the domain expert can make a choice from the Knack menu that adds an additional example value to the domain model and to the set of appropriate values for the aperture name variable, in this case, "environmental control panel." In this manner, the domain expert variablizes each report fragment in the sample report. The expert system now has report text that presumably (depending on how astute the domain expert is in generating robust text that pertains to the majority of the weapon systems under consideration) is appropriate for a particular phase of the weapon system acquisition cycle.

At this point, the domain expert has constructed all the structural elements of the expert system. That is, he/she has generated the domain

model that contains the declarative knowledge and relationships among the objects in his(her) domain of expertise. He/She has variablized a report text that is appropriate to reporting requirements. Still missing is the means by which the expert system obtains information from its user. Knack allows the domain expert to choose from among several strategies for obtaining this information. These strategies are then built into the expert system through interaction between the domain expert and Knack. The process is similar to that for the sample report fragments.

Strategy Selection

Knack automatically prompts the domain expert for a sample strategy for each object-characteristic variable that is variablized during the report fragment variablization process. In addition, the expert can explicitly select from a set of menus any object/characteristic variable as the target for a strategy selection. This allows him to select strategies for any object/characteristic variable that appears in the domain model. Knack allows the domain expert to choose from among the question, inference, formula, menu, and graphics strategies that the expert system uses to solicit information from its user. It is envisioned that Knack will eventually include database and table strategies as well. When the domain expert chooses the question strategy for the value of an object-characteristic variable, Knack prompts him(her) for the text of an appropriate question that solicits information from the expert system user. The domain expert constructs these questions keeping in mind that he/she can use example domain model values in the text that lead to condition elements of OPS5 rules during the variablization and instantiation editing process. Knack leads the domain expert through this process in the same manner as for report fragments. The result is a variablized question that will appear in the expert system user interaction at a time consistent with the status of information previously supplied and the condition elements on the left-hand side of the OPS5 rule that embodies the question to be asked.

The inference strategy allows the expert to infer a value for an object-characteristic variable on the basis of information previously supplied by the user. That is, any example value provided by the domain expert in the Knack domain model can be inferred during an expert system user interaction if the conditions prevail for this inference. These conditions are set by the domain expert during the Knack inference session. All the selection during this session is through menu interaction. The inference strategy is variablized, and instantiations are

edited by the domain expert in the same fashion as for report fragments.

The menu strategy allows the domain expert to delineate a complete set of possible values for an object-characteristic variable from which the user can choose during an interaction. The domain expert formulates the text of the instructions for menu use and the values that appear in the menu by using the strategy construction menus provided by Knack. This method is preferred when the possible values are single English words, simple English phrases, or numeric values and if the domain expert can enumerate a complete set of such values. Variablization of the text and editing instantiations provided by Knack are carried out as previously described.

The graphics strategy is implicitly carried out in Knack by coordinating the expert system kernel code that contains a graphics module with the Knack domain model. This graphics module is written in C and allows the expert system user to input geometric coordinate and name values for specified objects. These objects are the most common objects that appear in the domain model and are specified by menus that appear during interaction with the graphics module of the expert system kernel (inference engine).

With the formula strategy, the domain expert constructs a calculational pathway among the domain model object-characteristic variables to evaluate the hardness of any system component to a particular nuclear weapon effect. The domain expert selects object-characteristic variables for which he/she constructs formulas (typically, rather simple algebraic expressions) involving other variables of the domain model. Knack provides a window and menu system by which the domain expert selects variables for the formula and an editing window within which he/she specifies the mathematical relationships among the independent variables of the calculation. Knack automatically generates the OPS5 code condition elements for the desired variable by using the dependencies on the other variables within the formula. The domain expert must assure that his(her) calculational pathway is complete to the end-point constraint, which compares two values, each resulting from some formula calculation. The testing of the specified constraint determines whether the component is hard or not hard to the subject nuclear weapon effect.

Constructing Constraints

The domain expert uses Knack to specify constraints on object-characteristic variables. These constraints are constructed by following a set of menus prompted by Knack. For example, figure 4 shows a constraint on the threat voltage characteristic of the equipment object in the domain

model. This constraint, <equipment.threat-voltage> <= <equipment.safe-voltage>, sets the limit for these variables so that the equipment can be judged hard to the nuclear weapon effect. (Here, the symbol <= means less than or equal to.) If the constraint is violated, that is, if the threat voltage is greater than the safe voltage, the equipment is not hard to the weapon effect. The domain expert designs all such constraints into his(her) evaluation module that are meaningful and necessary to the hardness evaluation (EMP, INR, blast, and thermal).

Suggesting Fixes

The domain expert interacts with Knack to suggest fixes for any constraint that can be violated by responding to menu choices derived from the domain model. The expert has presumably established the fix types by constructing a node for each type in the domain model. In addition, the expert can set priorities among the suggested fixes, thus establishing the order in which a set of fixes is suggested to the user.

Results

The Wringer expert system prototype developed by the CMU team has been tried with 16 different users. The resulting reports have shown productivity improvements as much as 30 times greater than with previous methods. A report generated by previous methods required 30 days or more to complete, but an acceptable report was generated by these users within 1 to 3 days. HDL staff members who supervise the hardness program for the U.S. Army have given such reports an A- in terms of quality compared to a C for previously submitted reports. This productivity and quality improvement is expected to persist throughout the 6 Wringer expert systems that are required over the life cycle of a weapon system.

Users who have tried the prototype Wringers have ranked in the 50- to -95-percent range of professional nuclear effect hardening experts. Those who are less proficient increase their proficiency level by using the EMP evaluation module of the design parameters report. Those who have proficiency greater than Wringer uniformly comment on the solid benefit derived from using the expert system to assure the uniformity of the reporting and evaluation by vendors to the prime system contractor.

Summary

I described an application for expert systems that takes advantage of their capability to improve productivity and carry out technology transfer. This application is ideal for the use of expert systems. There is a vast need for routine reporting on the evaluation of the hardness of weapon systems to the effects of nuclear weapons. At the same time,

there is a shortage of experienced nuclear hardness design engineers. The application described here assures that expert systems can be developed that ameliorate the critical shortage of talent. The six expert systems developed for DNA and HDL show promise of productivity improvements by a factor of 10 to 30.

Acknowledgments

S-Cubed, a Division of Maxwell Laboratories Inc. provided technology transfer support and Booz-Allen & Hamilton provided nuclear weapons effects domain knowledge support.

References

Atkinson, G. 1989. Technology Transfer Utilizing Expert Systems. Presented at the Nuclear Defense Nuclear Agency Scientific Computing Symposium, Menlo Park, Calif., 11–13 July.

Atkinson, G., and Proffer, W. 1988. Expert Systems at Work for Harry Diamond Laboratories, Nuclear Survivability, Defense Nuclear Agency.

Boose, John H. 1988. Knowledge Acquisition Techniques and Tools: Current Research Strategies and Approaches. In Proceedings of the International Conference on Fifth Generation Computer Systems, 1221–1235. Tokyo: ICOT.

Chandrasekaran, B. 1983. Towards a Taxonomy of Problem Solving Types. *AI Magazine* 4(1): 9–17.

Department of Defense. Acquisition of Nuclear Survivable Systems, 4245.4, U.S. Department of Defense.

Klinker, G.; Boyd, C.; Genetet, S.; and McDermott, J. 1987. A Knack for Knowledge Acquisition. In Proceedings of the Sixth National Conference on Artificial Intelligence, 37–45. Menlo Park, Calif.: American Association for Artificial Intelligence.

Leaman, Claire M. 1989. "Rule-Based Structural Design in C. *AI Expert* May 1989:28–34.

Lenat, D. B. 1983. The Role of Heuristics in Learning by Discovery: Three Case Studies. In *Machine Learning: An Artificial Intelligence Approach*, eds. R. S. Michalski, J. Carbonell, and T. Mitchell, 243–306. San Mateo, Calif.: Morgan Kaufmann.

Marcus, S., ed. 1988. *Automating Knowledge Acquisition for Expert Systems*. Boston: Kluwer.

Rowan, D. A. 1989. On-Line Expert Systems in Process Industries. *AI Expert* August 1989:30–38.