Transitioning Expert System Technology: Case Studies at the Navy Center for Applied Research in Artificial Intelligence

Laura C. Davis
Alan L. Meyrowitz
Navy Center for Applied Research
in Artificial Intelligence
Naval Research Laboratory
Washington, DC 20375-5320
davis@aic.nrl.navy.mil
alanm@aic.nrl.navy.mil

Randall P. Shumaker
Information Technology Division
Naval Research Laboratory
Washington, DC 20375-5320
shumaker@itd.nrl.navy.mil

Abstract

Within the Department of Defense, a climate of budget cuts, personnel reductions, and renewed emphasis on research coordination and collaboration among the Services makes effective transition of developed technology from the research laboratory to the operational military more critical than ever before. This paper describes three case studies in expert or knowledge-based system technology development and transition at the Navy Center for Applied Research in Artificial Intelligence. Although quite different in methodological approach, the systems share a common development philosophy -- to construct prototypes as specialized, reusable tools or shells rather than as simple technology demonstrations tailored to a particular application. The paper concludes with the identification and discussion of a number of critical issues in technology transition drawn from our experiences with and lessons learned from these efforts.

1. Introduction

Since its inception eleven years ago, the Navy Center for Applied Research in Artificial Intelligence (NCARAI) at the Naval Research Laboratory has been engaged in research and development specifically designed to address the application of artificial intelligence (AI) technology and techniques to critical Navy and national needs. The emphasis at NCARAI is on applied research with the specific aim of demonstrating the applicability and effectiveness of artificial intelligence methods to practical problems. As part of the Navy's corporate research laboratory, the NCARAI pursues an in-house AI research program that is the broadest in technological scope within the Department of Defense (DoD), ranging from basic research and exploratory development in natural language understanding and machine learning to advanced decision aids and intelligent systems including image understanding and adaptive control [1,2]. However, significant AI development is also performed by other DoD research centers [3], and in today's climate of budget cuts, personnel reductions, and renewed emphasis on research coordination and collaboration among the Services, effective transition of developed technology from the research laboratory to the operational military is more critical than ever before. Additionally, the end of the Cold War has sparked increased interest by DoD and national laboratories in seeking industrial partners for potential technology transition as well.

This paper focuses on three case studies in expert or knowledge-based system technology development and transition at NCARAI. Although quite different in methodological approach, the systems share a common development philosophy. In NCARAI's early years, two problem domains of particular interest to the Navy were identified as good targets for research in knowledge-based system technology: equipment fault diagnosis and classification problem solving under uncertainty. Each had the advantage of being of high interest to particular Naval system development communities as well as having relatively broad application potential, and both involved interesting research issues with a supportive sponsor. An investigation of these particular problem domains quickly led to the conclusion that none of the available commercial tools was fully satisfactory for these applications.

The Navy has purchased and used a substantial number of general purpose expert system development tools, and NCARAI has made use of such commercial tools wherever appropriate. While these tools are powerful and useful in many circumstances, they are not well suited to a number of situations of interest to the Navy, including the two problem domains identified above. In particular, the representational paradigms available did not provide means for temporal and spatial reasoning, nor did they facilitate causal modeling; additionally, they failed to

accommodate the implementation of alternative methodologies for dealing with uncertainty, and they were relatively inefficient computationally.

Early in the planning process the decision was made to build system prototypes in the form of domain-specific, reusable tools or shells rather than as simple technology demonstrations tailored to a particular application. This decision meant that the systems developed had to include not only appropriate features for knowledge representation and inferencing in the selected domain, but also well-developed knowledge acquisition and human interfaces, user manuals and documentation, with special attention to code quality. This development philosophy still prevails, and ongoing efforts such as those in machine learning are investing the additional time and effort required to follow this strategy.

The following two sections describe the first and second of the three case studies to be reviewed -- our most mature systems that address the problem domains identified early on as being of great interest to the Navy. In particular, section 2 discusses the Fault Isolation Shell (FIS), a pioneering expert system in the application of causal-based modeling to the diagnosis of multiple faults in electronic equipment. Then in section 3, we describe the Bayesian Reasoning Tool (BaRT), a framework for uncertainty management that uses belief networks and other Bayesian techniques to compute the impact of uncertain evidence in classificatory problem solving. Section 4 describes the third case study, which represents a more recent emphasis on adaptive software to address the knowledge acquisition bottleneck, that plagues the knowledge engineering of more traditional expert systems, and to enhance robustness as well. It discusses a machine learning system, SAMUEL, that uses powerful, adaptive search techniques called genetic algorithms to learn high performance decision rules from a simulation of the operational environment. In section 5, we identify and discuss a number of critical issues in technology transition drawn from our experiences with and lessons learned from these systems, and our conclusions follow in section 6.

2. Intelligent Fault Diagnosis

The troubleshooting and maintenance of complex electronic systems has been a critical military and industrial problem for a number of years, and has attracted considerable interest as a problem domain amenable to expert system technology. However, for a number of reasons early fault isolation systems employing straightforward rule-based approaches have not proved successful in coping with the troubleshooting demands of large-scale electronic systems in use in the Navy and elsewhere. First, with several hundred different systems currently in use in the Navy, many large and complex, it is infeasible in terms of time and cost to consider the independent development of a traditional expert system for each. In addition, for some systems and subsystems, there are no human diagnostic "experts" available from whom to capture troubleshooting expertise to embed in a set of associative rules in the traditional knowledge engineering process. Moreover, even when experts are available, it is impractical to attempt to unambiguously map symptoms to causes exhaustively for very large systems. Inevitable gaps in the knowledge base result in performance uncertainty, large ambiguity groups of potential fault sources, and diagnostic inflexibility. Since troubleshooting technicians for many of these systems tend to depend on the functional and structural descriptions found in the technical manuals of the systems and subsystems they attempt to maintain, simple rule-based architectures alone may be inherently insufficient for the task.

The Fault Isolation Shell (FIS) was developed to address these issues. FIS is a causal model-based approach to the fault diagnosis of electronic equipment [4]. It is able to diagnose accurately multiple faults using a qualitative behavior model of a complex analog/digital system without simulation. A FIS-based system can be used in a variety of diagnostic settings. A primary use is as an interactive technician's aid, in which the troubleshooter and FIS work together to isolate system faults. A key to this mode of operation is a mixed initiative interface to FIS, through which it can accept and integrate input and actions by the technician as well as recommend actions of its own when queried by the troubleshooter. A second important use of FIS, particularly in the Navy, is in conjunction with automatic test equipment (ATE). In an attempt to reduce costs by automating the maintenance and troubleshooting process as much as possible, the military has made a substantial investment in ATE for avionics fault diagnosis.

Typically, ATE stations make tests, interpret test results, and recommend replacements by using software test program sets (TPS) based on decision trees designed when the equipment was built. Unfortunately, the TPS are very expensive to develop and keep up-to-date, and have proven rigid and inflexible in operation. FIS can help address these problems in several ways. First, if the ATE station has sufficient computing power, FIS can run directly on the unit and act as an ATE controller to provide dynamic decision making. Alternatively, FIS can be used by test engineers to help generate the test program sets used by the ATE in their standard mode of operation, thereby significantly reducing their cost and increasing their efficiency.

The FIS system architecture is illustrated in Figure 1. Of particular note is the system's unit under test (UUT) knowledge acquisition capability, with a specialized interface by which test and design engineers, who may be quite familiar with the UUT at hand but not well versed in computer science, can describe the unit to the computer. Other specialized interfaces are provided for FIS system debugging, ATE test tree generation, and the

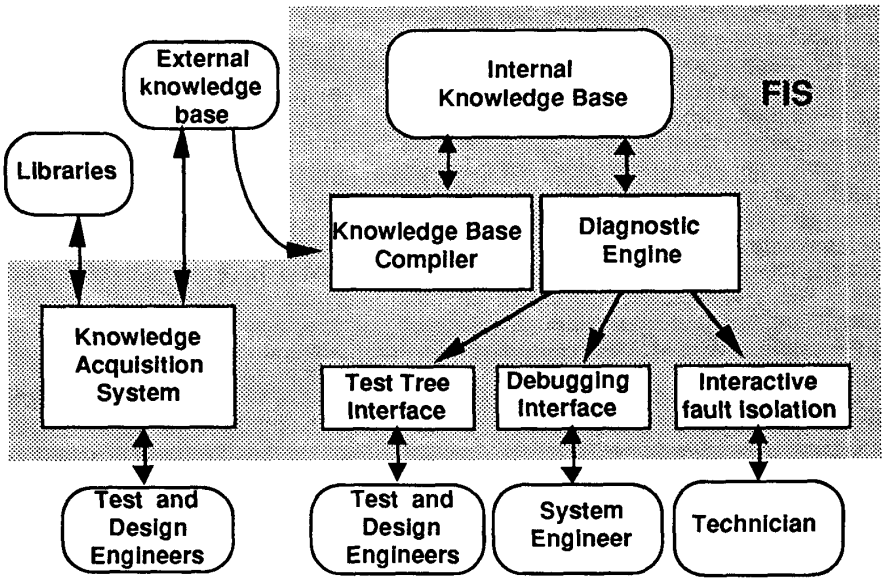

Figure 1. Fault Isolation Shell Architecture

troubleshooting technician. Through the interactive fault isolation interface during a troubleshooting session, FIS can update current beliefs about the UUT based on the technician's entry of test results, respond to such technician's queries as the probability of a fault hypothesis or the merit of a test, and recommend the next best test to make or component to replace. FIS also features efficient methods both for evidential reasoning specialized for device troubleshooting based on Bayesian principles and for computing the entropy of a complex system for use in best test selection. The shell is written in Lisp and runs under Sun Common Lisp on Sun SPARCstations.

System development and the initial implementation of FIS took approximately three years, with another year and a half devoted to rehosting, testing, evaluation, updating and refinement, and development of documentation and training materials [5]. The major time and effort devoted to preparation for transition of this technology is unusual for research projects, particularly in government, and is probably most similar to commercial software practices. In order to reach the application community NCARAI participated in several large test industry conferences, including a special session of IEEE AUTOTESTCON devoted to the subject of artificial intelligence in testing. As part of this process a display booth and on-line demonstration of FIS were created.

Evaluation of this domain-specific shell was also a concern. The sponsors of the effort in its latter stages organized a year-long beta test program involving five test sites. Each was provided with the FIS system resident in a transportable workstation to allow convenient evaluation. Each site was also furnished on-site training, manuals and other documentation, a working UUT example, and access to advice during the evaluation. The decision to provide the shell as a "turnkey" system resident in a computer was a direct result of experience gained during preliminary evaluation at several other sites. In these earlier evaluations most of the problems reported were installation- and operating system-related and secondary to system evaluation. Evaluations from the beta test sites proved useful and led to further refinements to FIS, and helped establish FIS as a forerunner to the now widely accepted causal model paradigm as state-of-the-art for intelligent fault diagnosis.

3. Reasoning Under Uncertainty

In dynamic military tactical environments, complex decisions must be made in short periods of time, with serious consequences for errors. Decision aids for use in such situations require rapid access to problem-relevant knowledge and inference procedures that produce acceptable solutions reliably. Of particular interest to the Navy are decision support systems for target classification problem solving. Crucial to these systems are efficient, reliable means for dealing with uncertainty. The uncertainty associated with sensor data and other evidence such as intelligence and archival information must be carefully accounted for to assure that inferences about the

implications of the evidence are plausible. Moreover, in military applications opponents are actively attempting to introduce uncertainty in the tactical environment, which also must be managed in the inference process.

Uncertainty is inevitable in dealing with real-world applications, but many of the methods currently used in expert systems are inadequate for military problems since they lack a sound, logical foundation for the representation, combination and propagation of uncertain data, or presume independence among evidence [6]. These computational approaches to uncertainty are often built upon the fundamental assumption that uncertainty can simply be "added on" to rule-based representations, implying that uncertain inferences can always be modularized in the same way as logical inferences. However, uncertain reasoning must often handle dependencies among hypotheses that are not modular, and accounting for these dependencies can be difficult using sets of modular rules of inference.

Another approach to this problem is to replace rule-based updating with an explicit representation of the relationships among hypotheses using probabilistic belief networks [7]. Such networks provide a graphical representation of dependencies among hypotheses. Each node in a belief network designates an uncertain variable; a probability distribution at each node characterizes at any time the belief for every potential value for that variable. Links between nodes use conditional probabilities to quantify how the belief in one node influences the belief in another. Since paths through the network summarize the direct and indirect relationships among hypotheses, belief networks provide a qualitative model of the inherent causal structure of a problem in uncertain reasoning. Furthermore, belief networks can be used as inference engines. As the information needed to update the belief distribution at a node is available locally from that node's neighbors, distributed, message-passing computations can be used to propagate the effects of changes in belief in accordance with probability theory and Bayes' rule [8].

The Bayesian Reasoning Tool (BaRT) is a generic toolkit for hierarchical reasoning that makes belief networks and other probabilistic techniques available for the management of uncertainty in classificatory problem solving [9]. Figure 2 illustrates the BaRT system architecture. As with FIS, BaRT provides a knowledge acquisition facility with a specialized interface, through which the knowledge engineer or network designer here is able to focus on constructing a single node at a time, quantifying the relationship between the current node and its immediate antecedents. BaRT provides a collection of canonical descriptions of probabilistic interactions that the network designer can instantiate for any node in the network, and allows the designers to maintain libraries of their own predefined subnetworks and canonical interactions that can be described once and stored for repeated use. In addition to Bayesian networks, BaRT also supports influence diagrams and taxonomic hierarchies as alternative knowledge representations, providing an

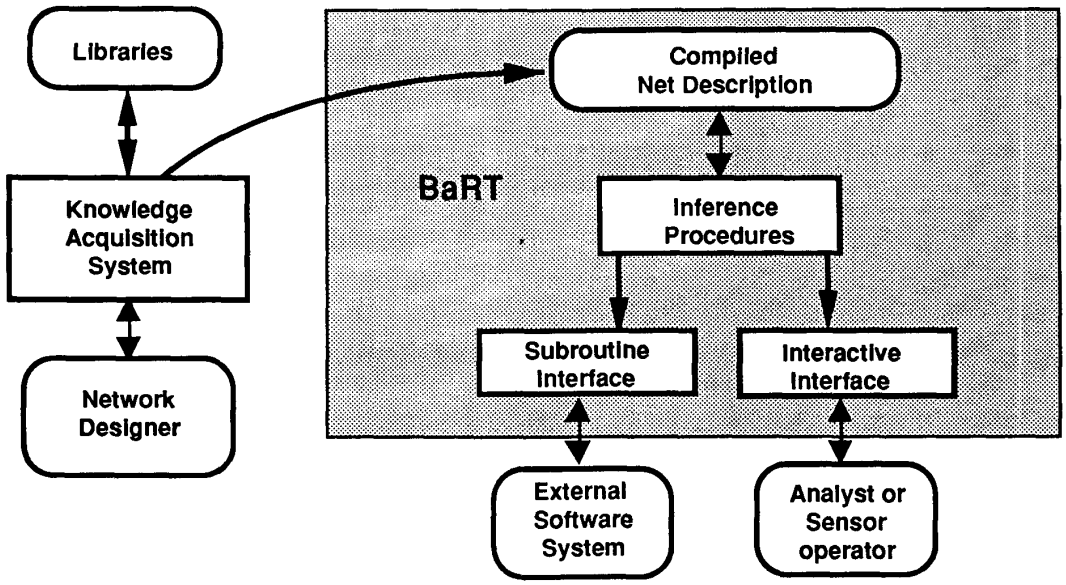

Figure 2. Bayesian Reasoning Tool Architecture

integrated capability for the system designer to "mix-and-match", selecting representations that best reflect the inherent structure of the problem being solved.

The inference methods in BaRT can be invoked in two ways. When BaRT itself is the primary problem solver, the analyst or operator communicates with the system through the interactive interface. An example of BaRT as a primary problem solver is an early implementation for interactive classification of ship images [10]. Far more prevalent, however, is for BaRT to perform inferences for some larger knowledge-based system that selectively makes use of its various belief maintenance capabilities through the subroutine interface. BaRT has been employed in this way in a system for situation assessment and target tracking [11], and is an integral part of a decision support system for tactical antisubmarine warfare battle management [12]. BaRT is written in Lisp and runs under Sun Common Lisp and X-windows on Sun SPARCstations, as well as on the older, Lisp-based Symbolics workstations.

BaRT has been tested in-house at NCARAI and informally by several outside users who obtained and embedded earlier versions within their larger systems. Although the interest in and desire for preliminary versions was gratifying, requests from users for help and advice, minor enhancements, etc., consumed considerably more time than anticipated and slowed ongoing BaRT development. However, since these carefully selected initial users had reasonable expectations and were knowledgeable systems designers themselves, the benefits of their experience with preliminary versions of BaRT, which included a number of excellent suggestions for system features and enhancements now implemented (as well as the identification of a few bugs now fixed), outweighed the liabilities of premature distribution [13].

4. Adaptive Software

In many problem domains of importance to the Navy, ranging from controllers for navigation by autonomous underwater vehicles to refinement of tactical doctrine, expertise can potentially be obtained directly through system problem solving experience, or precompiled expertise can be improved significantly based on actual system use. To produce software systems with adaptive behavior, researchers are developing machine learning techniques for automatic knowledge acquisition and system improvement through system experience or operation. A particularly interesting class of problems, sequential decision tasks, include the two Navy problem domains mentioned above. However, for many problems in this class, there exists neither a database of examples

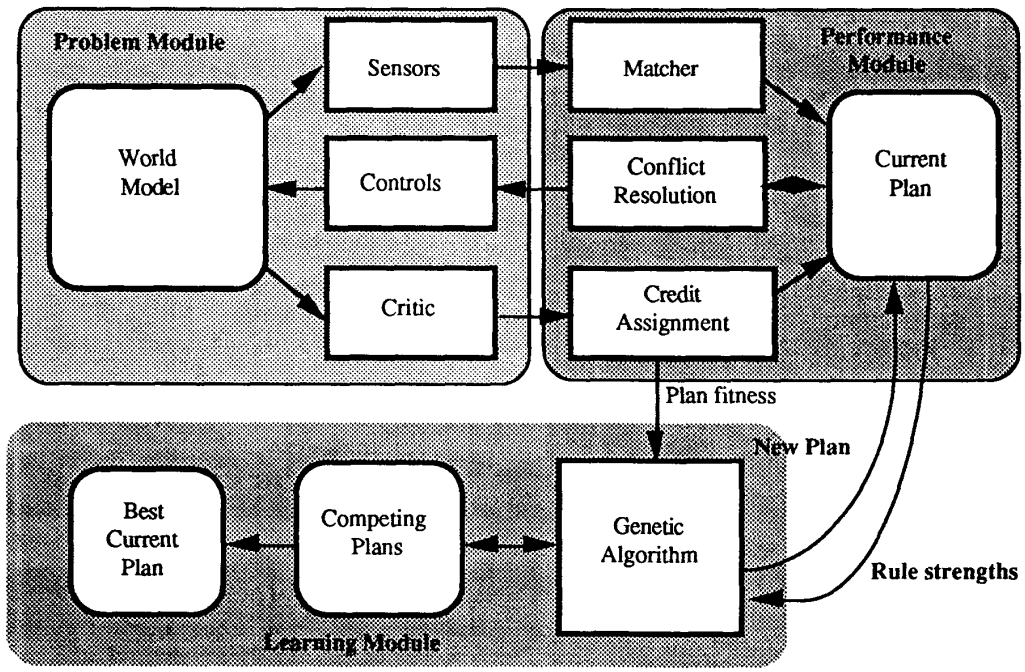

Figure 3. SAMUEL Learning System Architecture

nor a complete and tractable domain theory that might support traditional machine learning methods such as inductive learning or explanation-based approaches.

SAMUEL is a learning system shell that learns reactive behaviors, or strategies, for solving sequential decision problems [14]. It is based on genetic algorithms, a powerful search technique analogous to Darwinian evolution in which competition among strategies in the solution space is exploited to produce successive generations with improved performance [15]. SAMUEL is designed to interact with a task environment simulation or world model, in which it explores a range of agent behavior, and then uses feedback to revise its decision strategies over time. Strategies learned are sets of reactive production rules, the symbolic condition-action rules familiar to developers of traditional expert systems. An important aspect of the system is credit assignment, or more specifically the assessment of credit or blame to decision rules for an outcome [16]. This function is performed by the system critic module, which intermittently judges the agent's overall effectiveness at the task to be learned.

The SAMUEL system architecture is illustrated in Figure 3. SAMUEL has demonstrated in several different domains, including mine avoidance and local navigation, tracking, and evasion, that robust reactive strategies can be learned via simulation of the appropriate environment [17]. Previous studies have shown that knowledge learned under simulation may be applicable to the real world if the simulation is more general -- in terms of having more noise, more varied conditions, etc., -- than the real world environment [18]. A particularly intriguing application of SAMUEL currently underway with the Naval Coastal Systems Center is the adaptive testing of more traditional control strategies for autonomous underwater vehicles (AUVs) in simulation. Here SAMUEL plays the devil's advocate, probing for and then exploiting weaknesses in the AUV controllers by optimizing on those conditions that cause them to fail. SAMUEL is still very much a system under development; however, recent versions have been tested, fully documented [19] and distributed to selected users for informal testing, ever mindful of the lessons learned by the early distribution of BaRT. SAMUEL is written in C and runs under UNIX in X-windows on Sun SPARCstations.

5. Issues

Successful transition of expert system technology, as with any other marketable product, rarely happens by chance. From our experiences with these case studies we have identified a number of issues that influence the success or failure of technology transition from research prototypes to fielded systems. In this section we comment on each, relating lessons learned and our views on those issues yet to be resolved.

5.1 Design

Perhaps the most critical decision we have made relating to design in support of technology transfer was to adopt the philosophy of encapsulating our research developments in domain-specific, reusable shells. As related in our discussion of FIS, the initial system that tested this philosophy, much more time and effort than anticipated was required to provide the necessary software enhancements to support this strategy; in particular, much effort went into the knowledge acquisition modules and interfaces. However, our experience has been that these components can "make or break" the shell in the view of the design engineer using the tool to build a system application. We have received very favorable feedback, for example, on the knowledge acquisition component for BaRT, because of its support for implementing belief networks, even when the problem of management of uncertainty was not a major issue. Not surprisingly, the capabilities that the shell provides for the system application end-user interface are also an important factor. A current research effort at NCARAI is building a natural language interface to an expert system shell. Called InterFIS because of its initial application to FIS, this shell accommodates both typed and spoken natural language input and output and has attracted considerable interest among system design engineers [20].

5.2 Documentation

As with any research laboratory, published papers and technical reports as well as presentations at conferences, symposia, and workshops are important ways to document research progress and exchange ideas, and such activities are encouraged at NCARAI. However, for successful software technology transition, careful attention must be given to code documentation and particularly to user guides. Each case study reviewed earlier involved the production of at least one version of a user's manual. Researchers would much rather spend their time on technical papers than on user's manuals, but a well-written manual along with an example software shell application can go far to support a successful transition of good technology, or in the terms of the considerably more experienced commercial software world, a good market share for a desirable product.

5.3 Distribution and Marketing

Conference and workshop forums, as well as program reviews sponsored by the Office of Naval Technology, have provided opportunities for NCARAI to inform potential users of the availability of developed software tools. In addition, enterprising groups such as the Minnesota Project Innovation Inc. (MPI), a high technology small business development center supported

by the U.S. Small Business Administration and the State of Minnesota, have instituted a series of technology transfer symposia [21]. As part of its federal technology transfer program, MPI invites researchers from government laboratories to speak about their developed technology at focused symposia to an audience of small, high-technology oriented Minnesota businesses with potential interest in technology acquisition. However, with expanding distribution of a research product comes increased demands on the developer for assistance in answering questions on product functions and adaptation. Consideration has to be given to dedicating personnel to such technology transition activities or seeking support from a contracted intermediary to work with recipients of the products. The National Aeronautics and Space Administration (NASA), for example, has established the Computer Software Management and Information Center (COSMIC), to provide industry, other government agencies, and academia access to software technology developed for NASA projects. Operated for NASA by the University of Georgia, COSMIC currently has an inventory of over 1200 computer programs, which are available, with documentation, for a moderate fee. COSMIC maintains and advertises its software through two catalogs: one for those programs available for use within the United States [22] and another for those available internationally . NCARAI is in the process of evaluating the merits of employing such a service for its own software distribution. (Software distribution through COSMIC, for example, is available to any member of the federal laboratory consortium; however, less than 10% of the programs currently available through COSMIC originated outside of NASA-sponsored projects.) However, it has become apparent that there are unresolved legal issues to be addressed in going this route -- issues regarding patents, licenses, and liability.

5.4 Maintenance, Support, and Training

As a research laboratory producing technology prototypes, NCARAI is not in a position to engage in extended maintenance and support of its developed software. On a case-by-case basis, particularly while the development of the technology was still underway, we have provided limited support to selected recipients. Collaborative efforts have provided opportunity for more extensive support as the technology is applied to particular problems; however, this support is necessarily limited to the application at hand. Early efforts in the case of FIS to transition the technology for advanced development (and subsequent maintenance and support of the shell) to a Navy mission-oriented laboratory were sound in theory and had promising beginnings, but were eventually thwarted by funding difficulties at the target site. Initial activities included FIS training programs for target site personnel that ranged from two-day workshops to temporary detail to NRL for several months.

5.5 Enhancements and Extensions

As mentioned in the previous section, collaborative efforts with others, including those involving a challenging application of the developed technology, offer some opportunity for shell extensions and enhancements as well as maintenance and support. Perhaps the most promising avenue for enhancements and extensions, as well as longer range maintenance and support for such shells, is a particular type of collaborative effort -- the cooperative research and development agreement [23]. In 1986 Congress passed the Federal Technology Transfer Act to improve the competitiveness of U. S. industry by promoting the transfer of technology from government laboratories to the private sector. The CRDA provides a vehicle for cooperative research between a federal laboratory and an industrial corporation, another federal agency, unit of state or local government, university, or other nonprofit organization, with the intent that the non-federal-lab CRDA partner will pursue the commercialization of the technology. Obviously, there are a number of legal issues to be solved in entering such an agreement, including intellectual property rights, patents, licensing, and liability. Although NRL has entered into several successful CRDAs with industry, none involves software technology. Indeed, it is for software technology and in particular for knowledge-based methodologies that the legal issues appear the most formidable. However, as the legal profession gains more experience in dealing with these issues, guidance should become available in executing such agreements.

6. Conclusion

The NCARAI experience in transitioning expert systems technology has made clear the need to anticipate and address concerns along a number of dimensions: technical, administrative, and legal. In particular, the amount of time and effort required to identify and then assist potential technology recipients can easily be underestimated. Problems that arise once systems exchange hands can be particularly time-consuming in the areas of porting to new hardware, explaining and supplementing documentation, and training personnel through formal demonstrations or informally responding to queries. Research scientists who initially provide the technology may not be the best persons to oversee the transition process. On the other hand, they can be expected to benefit from user response in terms of obtaining important feedback on system limitations which can translate into exciting issues for further research.

References

[1] Shumaker, R., Davis, L., "Expert Systems at the Navy Center for Applied Research in Artificial Intelligence", *Expert Systems with Applications*, **1**, 71-77, 1990.

[2] Shumaker, R., "Expert Systems Technology Development and Distribution Experience at the Naval Research Laboratory", *Proceedings of the 1991 World Congress on Expert Systems*, Orlando, FL, 1991.

[3] Franklin, J., Davis, L., Shumaker, R., Morawski, P., "Military Applications", in S. Shapiro (Ed.), *Encyclopedia of Artificial Intelligence* (Vol. 1), New York: Wiley, 1987.

[4] Pipitone, F., DeJong, K., Spears, W., Marrone, M., "The FIS Electronics Troubleshooting Project", in J. Liebowitz (Ed.), *Expert System Applications to Telecommunications*, New York: Wiley, 1988.

[5] Spears, W., Pipitone, F., Marrone, M., *The Fault Isolation System User's Compendium*, NCARAI Document, Naval Research Laboratory, Washington, DC, 1989.

[6] Hamburger, H., Booker, L. B., "Managing uncertainty in expert systems: Rationale, theory, and techniques", in J. Liebowitz & D. DeSalvo (Eds.), *Structuring Expert Systems: Domain, Design, and Development*, 241-271, Englewood Cliffs, NJ: Prentice-Hall, 1989.

[7] Pearl, J., *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference*, San Mateo, CA: Morgan Kaufmann, 1988.

[8] Booker, L. B., "Managing Uncertainty in Target Classification Problems", *1988 NRL Review*, Naval Research Laboratory, Washington, DC, 1989.

[9] Booker, L.B., Hota, N., Ramsey, C.L., "BaRT: A Bayesian Reasoning Tool for Knowledge Based Systems", *Proceedings of the Fifth Workshop on Uncertainty in Artificial Intelligence*, American Association for Artificial Intelligence, Windsor, Ontario, Canada, 1989.

[10] Booker, L.,Hota, N., "Probabilistic reasoning about ship images", in J. Lemmer and L. Kanal (Eds.), *Uncertainty in Artificial Intelligence 2*, Amsterdam: North-Holland, 1988.

[11] Morawski, P., "A Hybrid Approach to Target Tracking", *Proceedings of the AI Systems in Government Conference*. Washington, DC, 1989.

[12] Mifflin, T., "Tactical ASW Battle Management System (TABS)", presented at the *Eighth Annual Conference on Command and Control Decision Aids*, National Defense University, Ft. McNair, Washington, DC, 1991.

[13] Hota, N., Ramsey, C., Chang, L., Booker, L., *BaRT Manual: Version 3.0*, NRL Memorandum Report 6778, Naval Research Laboratory, Washington, DC, 1991.

[14] Grefenstette, J. J., Ramsey, C. L., Schultz, A. C., "Learning sequential decision rules using simulation models and competition", *Machine Learning*, **5**, 355-381, Kluwer Academic Publishers, 1990.

[15] DeJong, K., "Genetic-Algorithm-Based Learning", in Y. Kodratoff and R. Michalski (Eds.), *Machine Learning* (Vol. 3), Morgan-Kaufmann, 1988.

[16] Grefenstette, J. J., "Credit assignment in rule discovery systems based on genetic algorithms", *Machine Learning*, **3**, 101-120, Kluwer Academic Publishers, 1988.

[17] Schultz, A. C., "Using a Genetic Algorithm to Learn Strategies for Collision Avoidance and Local Navigation", *Proceedings of the Seventh International Symposium on Unmanned Untethered Submersible Technology*, Marine Systems Engineering Laboratory, University of New Hampshire, NH, 1991.

[18] Ramsey, C. L., Schultz, A. C., Grefenstette, J. J., "Simulation-assisted learning by competition: Effects of noise differences between training model and target environment", *Proceedings of the Seventh International Conference on Machine Learning*, Austin, TX: Morgan Kaufmann, 1990.

[19] Grefenstette, J. J., Cobb, H. G., *User's Guide for SAMUEL, Version 1.3*, NRL Memorandum Report 6820, Naval Research Laboratory, Washington, DC, 1991.

[20] Perzanowski, D., Potter, B., "InterFIS: A Natural Language Interface to an Expert System Shell", *Proceedings of the 1991 World Congress on Expert Systems*, Orlando, FL, 1991.

[21] Minnesota Project Innovation Technology Transfer Symposium Series: Accessing Government Technology workshops, Minnesota Project Innovation Inc., Minneapolis, MN, 1991.

[22] *The COSMIC Software Catalog*, Computer Software Management and Information Center, The University of Georgia, Athens, GA, 1992.

[23] *Technology Transfer through Cooperative Research and Development Agreements*, NRL Publication 185-1003, Naval Research Laboratory, Washington, DC, 1991.