Int. J. Man–Machine Studies (1987) **26**, 213–230

INFORM: an architecture for expert-directed knowledge acquisition

ERIC A. MOORE

Applicon/Schlumberger, 4251 Plymouth Road, Ann Arbor, MI 48105, U.S.A.

ALICE M. AGOGINO

Department of Mechanical Engineering, University of California, Berkeley, 5136 Etcheverry Hall, Berkeley, CA 94720, U.S.A.

This paper presents an architecture for INFORM, a domain-independent, expert-directed knowledge acquisition aid for developing knowledge-based systems. The INFORM architecture is based on information requirements and modeling approaches derived from both decision analysis and knowledge engineering. It emphasizes accommodating cycles of creative and analytic modeling activity and the assessment and representation of aggregates of information to holistically represent domain expertise. The architecture is best suited to heuristic classification problem-solving (Clancey, 1985), in particular domains with diagnosis or decision-making under uncertainty. Influence diagrams are used as the knowledge structure and computational representation. We present here a set of information and performance requirements for expert-directed knowledge acquisition, and describe a synthesis of approaches for supporting the knowledge engineering activity. We discuss potential applications of INFORM as a knowledge engineering aid, specifically as an aid for developing insight about the encoding domain on the part of its user.

1. Introduction

Hindrances to widespread application of expert systems include what are typically significant allocations of resources, of critical personnel (the expert) and of knowledge engineering effort and equipment. The knowledge engineer's efforts to replicate the knowledge underlying expert performance through encoding techniques that maintain the form of that knowledge are known as *knowledge acquisition*; the design of tools and techniques to manage and support the process, as well as the active guidance of the process, is known as *knowledge engineering*. Knowledge acquisition is by far the hardest and most time-consuming part of the expert-systems building problem.

"Knowledge-acquisition bottleneck" understates the significance of the effort required to assess from a domain expert the information necessary to achieve expert performance. The resources required to build an expert system seem to have funneled the application of knowledge-based technology to only high payoff projects, involving only experts with highly valued skills. Here, the more specialized the expertise, and the more significant the application, the harder it is for someone outside of the expert's domain—the knowledge engineer—to build a system to replicate it. "Knowledge-acquisition Klein Bottle" might be more appropiate.

What can one do? We could relax the performance requirement, and settle for a knowledge-based system without expert performance, or reduce the scope of the

0020-7373/87/020213 + 18$03.00/0

target problem-solving domain, and settle for less functionality. Neither is likely to result in the most effective use of development resources. We could find persons with familiarity or proficiency with both knowledge-engineering tools and representation and the encoding domain (Fox, Lowenfeld & Kleinosky 1983), but even these persons are a scarce organizational resource. We could find less skilled persons in the domain that are likely to be more articulate about their problem solving (Dreyfus & Dreyfus 1980), but there is no assurance that these people will share their conceptual structure of the domain with that of the expert. Perhaps we could eliminate the expert knowledge engineer, and look for a way to let the expert encode directly.

The thought of having experts encode their expertise is compelling. Without an intermediary between the expert and the system, there is less noise introduced to the encoded knowledge, there is no time spent for the knowledge engineer to learn the language and concepts of the domain, and the resultant system has the expert's—and not an intermediary's—view of the domain (Friedland, 1981). For this, one risks losing process efficiency, for the expert must understand the knowledge representation and learn how to use the tool, one risks a potential loss of transparency, if the expert must recast his or her thinking in the tool's terms, and one risks failing to address objectively and fundamentally the expert's reasoning in the domain.

Like many established engineering organizations, the U. C. Berkeley Mechanical Engineering Department has many potential applications for knowledge-based technology, rich areas of domain expertise, and many senior and articulate experts, but it lacks readily available organizational knowledge engineering expertise, as well as tools demonstrably appropriate for the potential applications.

There is a very strong motivation to develop not just a toolkit, but a procedural aid, that will allow, for example, Master's level engineering students to successfully and efficiently employ knowledge engineering techniques and technology for practical problem-solving. Ongoing research has produced IDES, the Influence Diagram-Based Expert System, for doing probabilistic inference and planning using influence diagrams (Agogino, 1985, Agogino & Rege 1986] and (Rege, 1986a). This paper presents an architecture for INFORM, (INFluence diagram FORMer), an expert-directed knowledge-acquisition aid and interface for building knowledge-based systems in IDES.

2. Prior work on knowledge acquisition

We draw a distinction between *techniques, tools* and *aids*. A technique is a set of procedures, heuristics, or guidelines for performing Knowledge Acquisition (KA) or Knowledge Engineering (KE). A tool provides software support for application of the techniques, but no guidance on its own; knowledge engineers use tools. An aid is a tool that provides process guidance on its own. A domain expert undertaking any phase of a knowledge engineering project requires an aid.

KE tools, techniques, and aids in the literature address different areas of the knowledge engineering process: *encoding context,* the phase of determining how the characteristics of the domain, the expert, the user, and the application will affect or constrain KA procedure; *knowledge structuring,* the process resulting in an initial description of the knowledge base in the computational representation; and *knowledge refinement,* the process of model focusing and validation. Our research focus here is to provide tools that support KA in the context of developing techniques and aids for KE. Early work in KE was concentrated on developing tools and representations. The concept of the "knowledge level" (Newell, 1982), seeking to formally describe domain knowledge and problem solving at a level independent of implementation, has influenced the development of ontological representations of different problem solving domains, (Clancey, 1985; Alexander, Freiling, Staley, Rehfuss & Messick 1986), and languages specialized to problem-solving types (Bylander and Mittal, 1986). Recent emphasis has been on methodologies for structured KA (Freiling, Alexander, Messick, Rehfuss & Shulman 1985; Kline & Dolins, 1986 deGreef & Breuker, 1985), on assuring that the KA process meet the communication requirements of the application's organization (Grover, 1983), and on aids for rule refinement, (Kahn, Nowlan & McDermott 1985; Eshelman & McDermott 1986; Ginsberg, Weiss & Politakis 1985; Langlotz, Shortliffe & Fagan, 1986].

The organizational and structured knowledge-acquisition approaches are information-driven in the sense that they are a formalism, a set of activities, which produce documents and assure that information requirements and checks are met. These approaches emphasize building a paper knowledge base, or building a conceptual or knowledge-level structure of the domain, before committing programming resources; here, experts can describe their domain structure in some accessible representation freed from the implementation representation and with minimized reformulation by the KE. The KE is later involved, however, in rule encoding and refinement.

De Greef and Breuker (1985) see two basic approaches to knowledge engineering: the skills/programming-based rapid prototype and test approach (Hayes-Roth & Waterman, 1983; Brownston, Farrell, Kant & Martin, 1985) and the structured knowledge-engineering approach, which guides and supports an initial knowledge-acquisition phase while implementation is deferred. The INFORM architecture actually falls between the two; we employ model refinement techniques from decision analysis and knowledge engineering in an environment that is predominantly structured knowledge acquisition.

Successful KA aids for domain dependent systems in both KE and Decision Analysis (DA) exist; their design typically provides a domain-based encoding language and set of problem solving primitives, domain specific graphics, or some superset of domain concepts from which a temporal encoding problem will be identified (Holtzman, 1985; Musen; Fagan, Combs & Shortliffe, in press; Merkhofer, Robinson & Korsan 1979). "Domain-independent" implies that, for a given problem-solving approach, the user must create concepts, rather than select them, or that many meta-models of domain concepts are included in the tool model. Two aids for knowledge structuring, ETS (Boose 1984, 1985) and ROGET (Bennett, 1983) elicit the expert's structure of domain concepts though sequences of comparisons among sets of proposed objects. Both are intended for use by domain experts and result in "executable" rule bases. ROGET aids the user in choosing the appropriate inference technique and ontological representation, given information about the user's experience and the problem-solving type (as subsets of the classification problem-solving model). INFORM is intended to be domain independent across the range of heuristic classification problem-solving, but is potentially adaptable to specific domains.

3. Expert-directed knowledge acquisition

The notion behind "expert systems" is the desire to replicate an expert's problem-solving performance in a domain. While rule- and frame-based expert systems are proving to be effective computational representations of knowledge and expertise, they are not complete cognitive models (perhaps not even cognitive) of that knowledge or expertise. So the process of capturing knowledge, of transferring the expert's cognitive structures, representations, and methods to computational domain structures, knowledge representations, and procedures, will almost certainly entail its reformulation. If, for the expert, the act of articulating this knowledge to an audience is novel, then the expert is also reformulating his or her knowledge. Knowledge engineering is thus both a descriptive and creative modeling activity.

We view knowledge engineering as a model design and software engineering activity. A proportionally small amount of KE time is actually spent programming (Freiling *et al.*, 1985, Grover, 1983); much of the skills (and effort) of knowledge engineering are modeling skills—analysis and reduction, information management, and process decision-making—as well as the traditional emphasis on performance replication through incremental refinement. An expert-directed KA interface must support all of these activities to in turn successfully support a model's elicitation and eventual refinement.

The key assumptions behind any approach to self-encoding are that:

It is plausible that the expert can efficiently learn and use the encoding interface: that the expert understands how to use the tool, understands the problem, and is motivated enough to use the tool conscientiously;

The expert can think abstractly about the domain and problem-solving within it, i.e. identifying variables and influences;

A structured, analytic approach to thinking about one's domain knowledge and problem solving can achieve a refinable model;

The inevitable loss of transparency in encoded information is acceptable if the expert can somehow assure the performance of the model or if the expert is capable of thinking in the terms of the transformed model.

4. Decision analysis, influence diagrams, and knowledge-based systems

Decision analysis (DA) brings a body of experience to structured KA that meshes well with other approaches from within the AI community.

4.1. DECISION ANALYSIS

The *decision analysis cycle* (Matheson, 1977) is an iterative and interactive proscription for assuring that essential steps in the decision process or decision-making problem have been taken. It separates the process into deterministic structuring, probabilistic assessment, and informational phases. Assessment and modeling procedures direct the formation of choices, information, and preferences into the decision set.

Both practitioners of DA and KE face the problem of attention focusing, not in making analyses complicated enough to be comprehensive, but rather keeping them simple enough to be affordable and useful (Howard, 1980). DA structuring aids have taken a largely "top–down" approach to modeling a domain, and the KE aids a "bottom–up" approach to describing the relations in a domain based upon examples of problem-solving performance.

4.2. INFLUENCE DIAGRAMS

Influence diagrams (Miller, Merkhofer, Howard, Matheson & Rice 1976; Rege & Agogino, 1986*b*) are an intuitively attractive conceptual and operational representation for domain expertise. We use influence diagrams as a knowledge structure: a way of organizing knowledge that is operational, but that makes no cognitive claim. Influence diagrams have developed into a decision analysis tool that graphically represents the structure of the decision problem but maintains the computational utility of the decision tree (Shachter, 1985). They are a three-layered knowledge representation, consisting of information at three hierarchical levels: *relational, functional,* and *numerical.* This hierarchy accommodates well the way people tend to model from simple to complex, and from conceptual to numeric.

At the relational level, influence diagrams are directed acyclic graphs that represent the interdependence of uncertain events in a complex system. Nodes represent sets of possible events, or a range of properties for some object. The presence of an arc indicates the possibility that the outcomes of one node are somehow influenced by the outcomes of the other. At the relational level, they superficially resemble semantic nets and frames. A major distinction is that Bayes' theorem allows topological solution, or "re-orienting" of influence diagrams. Pearl's work with Bayesian networks (Pearl, 1985) uses inheritance in a frame-based system to propagate uncertainties in a structure that closely resembles influence diagrams, though without decision nodes.

The functional level is a specification of the type of relationship between nodes, or "how" a particular event or object influences another. The functional level is traditionally probabilistic, with quantitative relations compressed into the stochastic ones, but influence diagrams can readily accommodate fuzzy, logical, and other functional relations (Rege & Agogino 1986*c*). The numerical level is a quantitative measure of the "extent" of the relationship.

Figure 1 describes a diagnostician's model for a simple centrifugal pump. At the relational level, we can say that the pump's "discharge" is influenced by the "foot-valve state" and "stainer state". The likelihood that discharge is high, low, or nil, is influenced by the likelihood that the foot valve is open or closed and the likelihood that the strainer is clear, partially clogged, or clogged. At the relational level, we can specify that the arc from foot valve to discharge is "logical"; if the foot valve is closed, the discharge is nil. Or we could specify a probabilistic relation, and give a distribution on the probability of discharge being high, low, or nil, given some joint distribution of strainer and foot valve states. The diagnostic inference problem

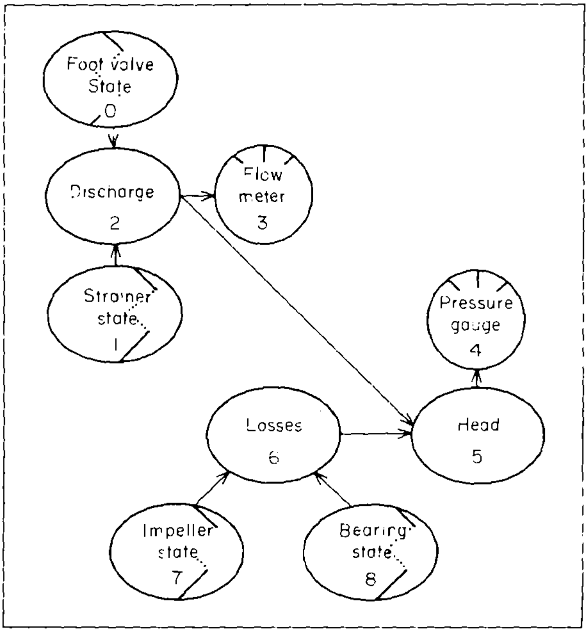

FIG. 1. Sample influence diagram.

is formulated as, for example: "given some flow meter reading X, and some pressure gauge reading Y, what is the probability that the strainer is clogged?"

4.3. BRINGING DECISION ANALYSIS TO KNOWLEDGE-BASED SYSTEMS

INFORM, because it is based on influence diagrams, is seen as best fitting applications under heuristic classification problem-solving (Clancey, 1985). The formal influence diagram representation is quite concise; there are nodes (a set of possible states for an event), states spanning the range of possible outcomes or values for the event, and probabilities on the occurrence of those states conditioned on other events. An arc in an influence diagram represents a heuristic link between a class of concepts in the domain. Data abstraction is subjective, rather than symbolic; the information lies in the uncertainty assessment or from further structuring, rather than in endorsements or in classification hierarchies.

Applying DA to knowledge-based systems means that we focus on designing problem-solving models that effectively replicate expert performance, rather than concentrating on implementing descriptions of that performance. It is important to separate replication of performance from duplication of procedure—at best, duplication is unlikely to result in performance improvement. Rather than implement actions emulating expert's problem solving actions, we want to use the expert's judgement to construct the model and to evaluate the model's performance.

Langlotz *et al.* (1986) point out one of the side benefits of doing first- and second-order sensitivity analysis on heuristics: the KE has to think more broadly about the concept, not just what its value is, but what it could be, and how likely those other values might be. Decision analysis is normally employed for significant non-routine decision-making where there is uncertainty about the state of the factors influencing the decision, the outcomes of the decision, or the extent to which the factors may influence the outcome. Knowledge-based systems are normally restricted to important but routine problem-solving, perhaps with the most frequency to heuristic classification style problems. In situations where it is uneconomical or impossible to replicate the expert problem-solving processes, the DA approach may be a viable way to approximate expert performance without explicitly relying on the processes behind it.

Barr argues for knowledge-based systems that provide insights, and not merely answers (Barr, 1985). He sees the largest measure of the utility of expert systems in the fact that their construction forces critical re-evaluation of one's own expertise. The same has been said of the utility of Decision Analysis (Howard, 1980). Non-transparency (reformulation, rather than replication of a true expert's problem-solving skills), represents a potential corruption of those skills, but can improve domain skills in non-experts, persons who would not ordinarily get the benefit of the KE's critical attention. Non-experts and experts alike may gain improvements through articulating, structuring, and recording for examination relationships and strategies in the problem-solving domain.

For knowledge acquisition and knowledge engineering, AI research and Knowledge-Based Systems practice offer:

information manipulation and management tools;
operational models of problem solving types;
passive and intelligent interface design concepts;
models of users and user problems;
prototyping/system development techniques;
tools and techniques for model refinement;
techniques for heuristic control;

and Decision Analysis offers:

Normative models for decision-making;
Robust techniques for modeling structure;
Practical encoding techniques for uncertainty;
Experience in organizational integration and acceptance.

We see particular appeal in bringing the top–down modeling and Bayesian uncertainty approaches of Decision Analysis and the influence diagram together with the software engineering tools and performance refinement techniques of Knowledge-Based Systems.

5. The information requirements for expert-directed knowledge acquisition

The design of an interface must be based upon the needs and abilities of the set of users for the set of tasks composing the application. However, the interface must also assure that it gets to the application the information it needs to run. In this sense, INFORM is a port for putting information into a program—subject to requirements for content, quality, and ease of expression.

There is no escaping the need to engineer information in order to represent knowledge. At issue, here, of course, is how best to give the expert some responsibility for knowledge engineering. INFORM is responsible not only for meeting the information needs of the computational knowledge representation, the influence diagram, but for meeting the information needs of a knowledge engineering process: context definition, model structuring, model refinement, and process decision-making.

In the context of assessment, an influence diagram is a framework for experimenting with a model's behaviour. The encoded diagram must represent information and must communicate an understanding. Much of this deeper information is descriptive, representing controlling assumptions, constraints on those assumptions and endorsement for or against them, intentions, histories, and alternatives. To communicate this understanding, we must represent information of different types: graph information, text, numerical, deterministic, logical, and uncertain.

There are three basic types of information INFORM must represent:

Model: the knowledge base;
Procedural: information revolving around the state, history, and direction of the KE process;
Insightful: information adjunct to analytic and creative thinking about and explanation of the model.

In this section, we discuss these information types in terms of their form and assessment.

5.1. MODEL INFORMATION

The information in the knowledge base is divided into *Computational, Structural* and *Uncertain* conceptual information types.

Computational model information. These are the representational requirements of the formal influence diagram. Nodes, states, probabilities, outcomes, and arcs map from a heterogeneous collection of C data structures to formatted matrices and probability distributions for topological transformation and numeric calculation within IDES.

Structural model information. An influence diagram represents a set of concepts and a way of associating related concepts. The underlying information may merely be descriptive to be useful. Augmented with context and assumption tags, and with their graphical representation, influence diagrams are an appealing way to structure the knowledge in a domain.

Uncertainty model information. Despite the naturalness of the influence diagram representation, both temporal and domain-acquisition problems are difficult for an expert or some other user to solve without experience or training and in some cases, without assistance. While Bayesian probability is a particular strength of the influence diagram, encoding for decision-making and diagnosis problems presents difficulty. Probability encoding is tedious. People's numeric estimates of uncertainty invariably do not accurately represent their underlying judgement without some structured revision and debiasing (Raiffa, 1970; Kahneman, Slovic & Tversky 1982). The process of encoding uncertain information may affect the values assessed and so is critical to the utility of that information (Spetzler & Holstein, 1977).

The many alternative uncertainty calculi are in part responses to these problems. The failure, however, of any one representation to win widespread acceptance as the "best" underscores the need for richer representations.

Bonissone & Tong (1985*a*) present further guidelines for assessment; these dovetail with what we already know from decision analysis to be important in terms of structuring the uncertain variable. Their discussion is valuable because it presents the uncertainty encoding activity explicitly as an information problem. So then, for each piece of evidence, one should determine the:

measure of certainty/uncertainty;
source of the evidence;
credibility of the source;
environmental conditions under which the source gathered information;
sensitivity of the goal to evidence;
cost of facility to gather information;
likelihood of succeeding in gathering information;
cost of this information gathering task;
default plan to accomplish this task.

The encoding of expert's uncertainty estimates is as least as important as the internal representation of that uncertainty in a knowledge base. One essential perspective on uncertainty representation which sometimes gets lost is that the representation must be intuitively agreeable to the expert—both the expert and the uncertainty representation must speak the same language. As Bonisonne (1985*b*) points out, it will ultimately take a mix of verbal and numeric representations to cover adequately the Babel of uncertainty representations used by different experts in different domains.

Influence diagrams are founded on Bayesian probability. Cheeseman (1985) argues that Bayesian probability, if properly used, can be worked around virtually all objections to it; in his view, the faults of Bayesian probability are based primarily on the misperceptions of its critics. On the other hand, a number is a rather sterile representation of a quantity that, cognitively, appears to be in large part verbal (Zimmer, 1985). A strictly Bayesian numeric estimate is very convenient, and axiomatically correct, but is often misleading without a complete view of the priors implicit in the assessment. Further, a single number overestimates the crispness of the state of knowledge about that uncertainty. A verbal assessment incorporates more factors than a numeric one, but computation, without loss of information, requires that the user's fuzzy functions be known as a context-specific mapping of verbal to possibilistic (Zimmer, 1985) or probabilistic numeric distributions. Evidential reasoning emphasizes articulating priors acting on an estimate and the decision-making power of simply ranking outcomes [much like the Analytic Hierarchy Process in decision-making (Saaty, 1980)]. All of these approaches, under some conditions, make a strong case for themselves. With influence diagrams, we are committed to representations that can ultimately be mapped to Bayesian probability.

Certainly a judgement on the strength or weakness of one representation or another should consider encodability of that representation. In assessing an uncertainty estimate, considering all the approaches, one would want to:

rank comparable outcomes in order of likelihood;
assess a verbal (qualitative) estimate;
assess numeric Bayesian values;
elicit underlying evidence for an assessment;
estimate the range of uncertainty;
elicit direct conditions on the validity of the assessment.

However, decision analytic information assessment calls for no less than all of this information. What we find is that, even though the computational representation may be considerably sparer, the conceptual representation must include an aggregate of information about the uncertain quantity. We contend that a well-designed encoding and representation environment can make the encoding of Bayesian probabilities for expert systems less forbidding and more accurate. Such an environment would support a composite conceptual representation of uncertainty (including linguistic, numeric, underlying and conditioning priors), a mapping from verbal to numeric, and from numeric to verbal, and a numeric Bayesian calculus.

For INFORM, the approach we will take is straightforward:

(1) first assess reference linguistic distributions in a broad context;
(2) use these linguistic assessments as a "first pass";
(3) for refinement, with more sensitive variables, or for variables misleadingly represented by the linguistic assessment, qualify the linguistic assessment for the specific context or employ traditional numeric encoding techniques.

5.2. PROCEDURAL INFORMATION

Supporting the KA process, for a self-encoding expert, or for some combination of KE and expert, requires information management tools (for representing the model and process state and history as basis for making choices about what to do next, and it requires guidelines and tools for making procedural decisions.

Many KA tools provide programming support, support interpreted incremental refinement, provide rule prompters, or a rule compiler based on a rule language. The DA framework is an approach that would complement all of these approaches. One view (Reboh, 1981) favors a system that requires the collaboration of the KE, but with techniques and tools for support of critical phases requiring little KE training. Such a system would in effect redistribute portions of the KE's expertise between the support tools and a domain expert or less skilled KE. This view is at the heart of the INFORM approach.

5.3. INFORMATION FOR INSIGHT

We regard "insight" as the creation and revision of a mental picture of the domain and processes within it, and the recognition and evaluation of possibilities and tradeoffs inside it. The modeling interface should provide the information and techniques for developing and maintaining insight about the model. Insight is supported by the ability to conveying timely information from the model to the user;

if it is not easy for the user to organize and convey appropriate information to the model, insight is impeded. The base issue is achieving a fit between user and interface. INFORM attempts to provide a familiar medium and acceptable stimulus for modeling effectively.

Essay writing is a metaphor for the "progressive formalization" (Holtzman, 1985) of a decision analytic model. A contention of the INFORM approach is that expository writing could accomplish (in effect) what the motivation and structuring phases are intended to accomplish in the probability encoding process (Spetzler & Holstein, 1977); give the user the opportunity to address critically the assumptions, intentions, and methods behind the model.

The "standard form" for writing an essay is a well-known and widely taught framework for expository thinking and discourse. As an encoding approach, it goes a step beyond simply writing rules into some subset of a natural or domain language. It puts relations into paragraphs and sections in support of a problem statement or thesis—the user thinks about components of the model in a structured sense, in the context of other model components, the modeling context, and the modeling goal.

In reviewing a knowledge base encoded through INFORM, the DA, expert, user, or KE is in a sense reading a story; from the contents of the KB, the reader is supposed to be able to piece together a problem-solving "narrative". Applications to be used with persons other than the encoding expert require explanations that are dependent on the line of reasoning, the model description, and the background of the users relative to that of the expert. A very critical element for KA is visualizing an audience for the application. Viewed this way, we can say that the "rules of journalism" apply here too. In this sense, the reader must know—and the interface must somehow assess *Who, What, When, Where, Why & How* for its concept and relation.

6. INFORM

These three performance goals capture the essence of what INFORM is intended to achieve as a KA aid:

(1) *sufficiency,* getting the encoded information right in terms of the influence diagram representation;
(2) *correctness,* avoiding and correcting conscious and unconscious misrepresentations of expert judgement; and
(3) *providing insight,* at minimum representing the correct information in a comprehendable form, at best completely capturing an expert's underlying model of the objects, relationships, and inferences in his domain.

6.1. ARCHITECTURE

There are four conceptual levels to the INFORM architecture. They build from satisfying information requirements to giving more sophisticated tools and advice for insight and finally to a system which could effectively tutor its user through the KE process.

The first level is to fill the diagram to sufficiency, through satisfying structural and computational constraints. The second is to employ the kinds of feedback that decision analysts and knowledge engineers employ: diagram drawing on demand, graphic feedback of distributions, ordinal, deterministic and stochastic sensitivity analysis, comfortable interaction language, and opportunity for and access to extensive explanation about the modeling process and the encoded model. The third is the "heuristic" approach, where the system provides hints and suggestions for encoding to the user based upon normative models of the encoding process and sticky points in the domain. Finally, with "expert aid", the system provides aid (hints, requests for explanation, reformulation) based upon encoding session information, normative models of the encoding process, and descriptive models of the problem domain and user's encoding style.

The INFORM user, in an encoding session, goes through a problem and session structuring module, and then a succession of editing and analysis phases. At any time, the user may get help about the syntax, options, or intent of the current phase, or comment about some aspect of the model or the modeling process, or review some graphic or textual aspect of the model. Figure 2 shows the main modules and editor sequence.

6.2. KNOWLEDGE STRUCTURING

There are two key ideas to INFORM's structuring and refinement approach:

(1) start modeling at the most general level of precision or specificity;
(2) increase specificity only for the best improvements in model performance.

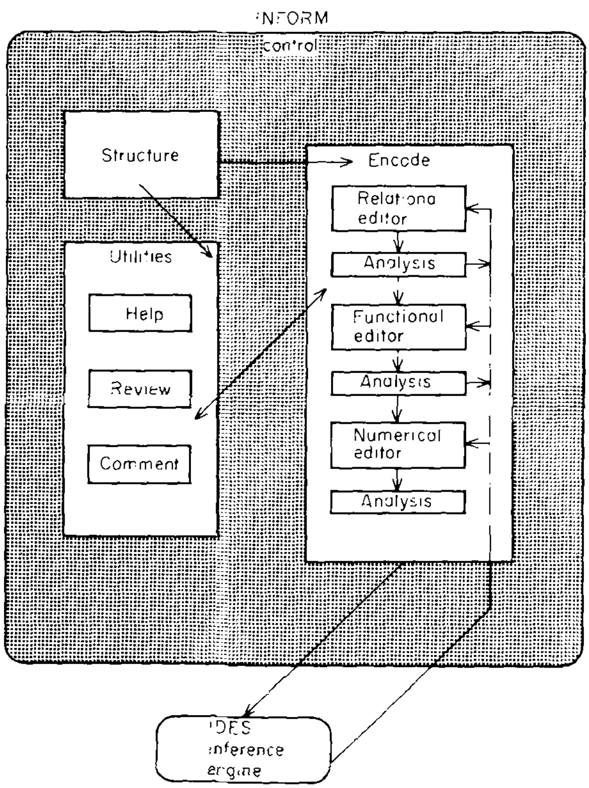

FIG. 2. Process paths and modules in the INFORM architecture.

The user is free to edit the model, and accept or reject advice on what task to choose next, but is guided through model analysis and refinement.

In the relational editor, the user specifies combinations of node name, node label, node description and arcs. On exiting the editor, this information is parsed and "incomplete information" is identified; the user is prompted to provide, for example, a description for a node identified only by a label.

The nodes operated on in the relational editor determine the nodes to be operated on in subsequent phases, in the functional and numerical editors. The analysis phases are directed by ordered lists: perform sensitivity analysis with the nodes the user is least confident about, expand the nodes that the outcome is most sensitive to, assess in a different way those uncertainty estimates the user is least (or most confident) about. The order of nodes operated on is determined by user ranking, or by some function of the rank of the node within the influence diagram.

Below are some of the activities in the INFORM architecture:

INFORM encoding activities

Set Context	System application? Encoding goal? Identify user group? Calibrate linguistic uncertainty? How much time is this encoding problem worth?
Model at Relational Level	
– Describe the Model:	Edit and compile nodes and arcs.
– Assure Completeness:	For each: name, label, description, givens, explanation.
– Look for Insight:	Potential modifications? Encoding plan?
– Offer Analysis:	Check for cycles, bushyness, sort objects by importance?
– Offer Advice:	"Might do this next"
Revise Model?	Does information about the state of "Node X" tell you something about the state of "Node Y"?
Model at Functional Level	
– Describe	Edit and compile functional form, states. Choose decision rule, or choose quantitative form or uncertainty representation?
– Assure Completeness:	Name, label, description, explanation, plan.
– Offer Analysis:	Estimate modeling effort?
– Offer Advice:	"Consider reducing the number of states in these nodes . . . "
Revise Model?	
Model at Numeric Level	
– Assure Completeness:	Assess aggregate uncertainty information.
– Offer Analysis:	Sensitivity, performance analyses.
– Offer Advice:	"Focus next on these most sensitive nodes . . . "
Revise Model?	

6.3. KNOWLEDGE REFINEMENT IN INFORM

"Refinement", in the DA context, is directed towards attention focusing, typically through ranking, and deterministic and stochastic sensitivity analysis, and towards balancing the modeling effort in terms of both structural granularity and value of additional modeling effort. Refinement in rule-based expert systems building is a process of rule addition and modification leading ultimately to performance replication. Performance improvement in a knowledge-based system generally comes with increases in specificity; because of the large assessment effort behind properly encoding probabilities, a good decision model will expand and contract through each refinement cycle. With influence diagrams, there is tradeoff between the granularity of the represented uncertainty and that of the model structure. Formally, influence diagrams rely on implicitly incorporating conditioning factors within the uncertainty assessment and in the concept's definition to result in a polished but condensed model. Rule-based expert systems representations, on the other hand, force this contextual information to be explicitly expressed as rules. Part of the refinement process in INFORM is the successive elaboration of what the model represents.

The success of our approach to encoding uncertainty during refinement is contingent on at least three things: that, given no new information, some consistency of uncertainty verbal to numerical mappings is maintained over time and domain; the success of linguistic revision given new contextual information; and the extent to which information from simulation and tests is incorporated into revised estimates.

6.4. INFORM IN THE FUTURE

The superstructure for INFORM has been implemented in C, and presently a single-display text and graph editor assesses relational information and automatically generates complex influence diagrams. We have written a linguistic calibration program; work is continuing at UC Berkeley with experiments to sample uncertainty vocabularies of graduate and undergraduate engineering students. Work is continuing with the re-implementation and development of the INFORM architecture at Schlumberger/Applicon with the Strobe/Impulse object programming and knowledge-base editing tools (Smith, Dinitz & Barth 1986; Smith, 1983).

The principal advantage of influence diagrams over decision trees is the explicit graphic representation of the interdependencies (or lack of) between events. Influence diagrams are fundamentally graphic entities; once a diagram has been created, the interface too should be organized graphically. Because we want to simultaneously represent different types of information about the model and the modeling process, a single view is inadequate. The interface under development will have static windows—for model graphics, model text, prompt window, editor/comments and "pop-up" windows—for agendas, menus, advisors, and uncertainty encoding and display. Given the need for a graphic representation, the interface should allow the diagrams to be created graphically (to be drawn on the screen), in addition to generating the graph from the user's entries of nodes and arcs.

INFORM is intended to be domain-independent; a specialization of the architecture could add to the interface the kind of checking rules that allow for domain and user dependent meta-knowledge about encoding. RACHEL (Holtzman, 1985) is one such system, a domain-dependent intelligent decision support aid for infertile couples.

Effective modeling approaches rely in part on the underlying domain and in part on the modeler's cognitive style. INFORM is a system intended to replace at least in part the expertise of the KE in directing and in giving advice to the encoder about the KE process, and in representing the encoded information. Implementation of the architecture may ultimately support active modeling and guidance of a particular expert's encoding effort. Prerequisites for such tutor include measurable or deducible standards of knowledge engineering performance and methodology for individual actors, and that these measures are conditionable on a fairly small set of inferable or directly assessable measures.

6.5. EVALUATION

We would like to test INFORM for absolute performance, as an interface and as a modeling tool, and for relative performance, against an encoding expert. The true test is application, taking an encoding problem from scratch with an acknowledged expert, and trying to build a working system in a nonlaboratory environment. The typical test of comparing system performance against that of an encoding expert is inappropriate at this point but ought to be an eventuality. At the interface level, we have used and continue to use "good design" checklists (Heckel, 1984), but that is no real assurance of a good interface without testing and experience. At this stage, while we are still developing an "integrated" system, user comments and ad hoc evaluation have been especially useful. More formal approaches to interface evaluation will be the next step.

There are further questions arising from the premises that we have based the INFORM architecture on. Are there correlations between level of skill and self-encoding ability? We would like to use INFORM in a group with measurably disparate levels of problem solving skill in a common domain. Does structured explanation support model articulation, or is this approach too much of a burden on the imagination and patience of the "typical" expert? Does having control over simulation and performance evaluation put the expert in a position where he or she is describing concepts and relations that exist near the problem-solving level, or will the expert still construct unrefinable models? These questions represent both sides of bets that we are making in this research.

A testing issue that is separable from the evaluation of the entire interface is the effectiveness and accuracy of our linguistic uncertainty encoding approach. Testing areas of interest include looking at differences in the language of uncertainty between estimates about uncertain events from inside and outside the encoding domain, consistency of judgements between subdomains, and the efficacy of different approaches to representing the encoding problem and conducting refinements.

6.6. POTENTIAL APPLICATIONS OF INFORM

When can we use a stand-alone aid such as INFORM for constructing a knowledge-based system or decision support system? We divide the spectrum of application problems down into significant and not-significant problems. Significant problems are "high stakes" problems, involving lives, or allocations of resources significant in the eyes of the sponsoring organization. For significant problems, in a novel domain, one would expect to be able to use a KA aid as a preprocessor (as in ETS) for initiating the model, but would certainly expect decision-analyst or knowledge-engineer involvement in model refinement and validation. Significant problems in a stable and well-understood context are liable to see the involvement of DA and KE, but it may not be always necessary. In both cases, the KA aid must accommodate well the involvement of DA and KE.

For problems whose solution is strongly driven by model structure, or where solution precision is not critical, INFORM is plausibly stand-alone. Other potential roles for INFORM in KE are as a "pre-processor", structuring the domain for explanation as a component of an intensive KA process; as a KA aid for supporting a novice knowledge engineer; as a personal KA tool for novel problems or where the modeling goal is oriented towards developing insight; as a domain-independent aid for non-significant problems; and as a domain-dependent aid for significant problems.

7. Conclusion

We feel knowledge-acquisition aids must support information assessment and presentation and must provide support for undergoing a sound modeling process. Fundamentally, the INFORM architecture is an aid for building models; it draws its knowledge structure and modeling approach from Decision Analysis, and its approach to handling information and heuristics about encoding from Knowledge Engineering. It is well suited for classification problem solving, especially under uncertainty. The support INFORM will provide for experts encoding is as a top down design aid, focusing on descriptions of the domain concepts and structure, rather than on examples of problem solving in the domain. Structure is edited, rather than induced. Such direct involvement of the expert in constructing an operational model of the domain we feel will aid *knowledge engineering for insight,* aiding the development of expert behavior not only on the part of the system, but on the part of the encoder as well.

This work was performed at the University of California, Berkeley, and was funded in part through grants from the National Science Foundation PYI Program, University of California's Project Micro, General Electric and IBM.

References

AGOGINO, A. M. (1985). Use of probabilistic influence in diagnostic expert systems. *Proceedings of the 1985 ASME International Computers in Mechanical Engineering,* Boston, MA, Vol. 2, pp. 305–310 (4–8 August 1985).

AGOGINO, A. M. & REGE, A. (1986). IDES: influence diagram based expert system. *Mathematical Modelling in Science and Technology Proceedings of the Fifth International Conference on Mathematical Modelling, 29–31 July 1985,* University of California, Berkeley.

ALEXANDER, J. H., FREILING, M. J., STALEY, J. L., REHFUSS, S. & MESSICK, S. L. (1986). Knowledge level engineering: ontological analysis. *Proceedings American Association for Artificial Intelligence 1986,* pp. 963–968.

BARR, A. (1985). Systems That Don't Understand. *Cognitiva: Artificial Intelligence and Neuroscience,* Paris, June 1985.

BENNETT, J. S. (1983). *ROGET: A knowledge-based consultant for acquiring the conceptual structure of an expert system,* Stanford HPP-83-24, October 1983.

BONISSONE, P. P. & TONG, R. M. (1985). Editorial: reasoning with uncertainty in expert systems. *International Journal of Man–Machine Studies,* **22,** 241–250.

BONISSONE, P. P. (1985*b*). *Summarizing Uncertain Information With Aggregation Operators (draft),* General Electric Corporate Research and Development, New York, 11 March 1985.

BOOSE, J. H. (1984). Personal construct theory and the transfer of human expertise. *Proceedings American Association for Artificial Intelligence* 1984, pp. 27–33.

BOOSE, J. H. (1985). A knowledge acquisition program for expert systems based upon personal construct psychology. *International Journal of Man–Machine Studies,* **23,** 495–429.

BROWNSTON, L., FARRELL, R., KANT, E. & MARTIN, N. (1985). *Programming Expert Systems in OPS5,* Addison–Wesley Publishing Company.

BYLANDER, T. & MITTAL, S. (1986). CSRL: a language for classificatory problem solving and uncertainty handling. *The AI Magazine* August, 66–77.

CHEESEMAN, P. (1985). In Defense of probability. *Proceedings International Joint Conference on Artificial Intelligence 1985.*

CLANCEY, W. J. (1985). Heuristic classification. *Artificial Intelligence,* **27,** 289–350.

DREYFUS, S. E. & DREYFUS, H. L. (1980). A five stage model of the mental activities involved in directed skill acquisition. *ORC 80-2,* U. C. Berkeley Industrial Engineering and Operations Research Department, February 1980.

ESHELMAN, L. & MCDERMOTT, J. (1986). MOLE: a knowledge acquisition tool that uses its head. *Proceedings American Association for Artificial Intelligence 1986,* pp. 950–955.

FOX, M. S., LOWENFELD, S. & KLEINOSKY, P. (1983). Techniques for sensor-based diagnosis. *Proceedings International Joint Conference on Artificial Intelligence 1983,* pp. 158–163.

FREILING, M., ALEXANDER, J., MESSICK, S., REHFUSS, S. & SHULMAN, S. (1985). Starting a knowledge engineering project: a step by step approach. *The AI Magazine,* **Fall,** 150–163.

FRIEDLAND, P. (1981). Acquisition of procedural knowledge from domain experts. *Proceedings International Joint Conference on Artificial Intelligence 1981,* pp. 856–861.

GINSBERG, A., WEISS, S. & POLITAKIS, P. (1985). SEEK2: a generalized approach to automatic knowledge base refinement. *Proceedings International Joint Conference on Artificial Intelligence 1985,* pp. 367–375.

DEGREEF, P. & BREUKER, J. (1985). A case study in structured knowledge acquisition. *Proceedings International Joint Conference on Artificial Intelligence 1985,* pp. 390–392.

GROVER, M. D. (1983). A pragmatic knowledge acquisition methodology. *Proceedings International Joint Conference on Artificial Intelligence 1983,* pp. 436–438.

HAYES-ROTH, F. & WATERMAN, D. A. (1983). An overview of expert systems, Ch. 1. HAYES-ROTH, F., WATERMAN, D. A. & LENAT, D. R. *Building Expert Systems.* Addison–Wesley Publishing Company.

HECKEL, P. (1984). *The Elements of Friendly Software Design.* Warner Books.

HOLTZMAN, S. (1985). *Intelligent Decision Systems,* Ph.D. thesis, Stanford Engineering-Economic Systems, reprinted by Strategic Decisions Group, March 1985.

HOWARD, R. A. (1980). An assessment of decision analysis. *Operations Research,* **28,** 4–27.

KAHN, G., NOWLAN, S. & MCDERMOTT, J. (1985). *MORE: an intelligent knowledge acquisition tool. Proceedings International Joint Conference on Artificial Intelligence* 1985, pp. 582–584.

KAHNEMAN, D., SLOVIC, P. & TVERSKY, A. (1982). *Judgement Under Uncertainty: Heuristics and Biases.* Cambridge University Press.

KLINE, P. J. & DOLINS, S. B. (1986). Problem features that influence the design of expert systems. *Proceedings American Association for Artificial Intelligence 1986,* pp. 956–962.

LANGLOTZ, C. P., SHORTLIFFE, E. H. & FAGAN, L. M. (1986). Using decision theory to justify heuristics. *Proceedings American Association for Artificial Intelligence 1986,* pp. 215–219.

MATHESON, J. E. & HOWARD, R. H. (1977). Introduction to decision analysis. In *Readings in Decision Analysis,* Strategic Decisions Group.

MERKHOFER, M. W., ROBINSON, B. & KORSAN, R. J. (1979). A computer aided decision structuring process. *Technical Report 7320,* SRI International, Menlo Park, CA, June 1979.

MILLER, A. C., MERKHOFER, M. W., HOWARD, R. H., MATHESON, J. E. & RICE, T. (1976). *Development of Automated Aids for Decision Analysis.* SRI International, Menlo Park, CA, 1976.

MUSEN, M. A., FAGAN, L. M., COMBS, D. M., & E. H. SHORTLIFFE (1987). Using a domain model to drive an interactive knowledge editing tool. *International Journal of Man–Machine Studies*. In press.

NEWELL, A. (1982). The knowledge level. *Artificial Intelligence*, **18,** 87–127.

PEARL, J. (1985). Fusion, propagation, and structuring in Bayesian networks. *Workshop on Probability and Uncertainty in Artificial Intelligence,* UCLA, 14–16 August 1985, RCA and AAAI.

RAIFFA, H. (1970). *Decision Analysis: Introductory Lectures on Choices under Uncertainty.* Addison–Wesley Publishing Company.

REBOH, R. (1981). *Knowledge Engineering Techniques and Tools for Expert Systems. Linkoping Studies in Science and Technology No. 71,* Software Systems Research Center, Linkoping University, *S-581 83,* Linkoping, Sweden.

REGE, A. & AGOGINO, A. M. (1986*a*). Sensor integrated expert system for manufacturing and process diagnostics. *Proceedings of the Symposium on Knowledge-Based Systems,* ASME Winter Annual Meeting, Anaheim, CA, 7–12 December 1986.

REGE, A. & AGOGINO, A.(1986*b*). Representing and solving the probabilistic inference problem in expert systems. *Proceedings of the ICS-86, International Computer Symposium,* 15–19 December 1986, Tainan, Taiwan.

REGE, A. & AGOGINO, A. (1986*c*). Fuzzy influence diagrams. *UC Berkeley Expert Systems Lab Working Paper 06-86-01.*

SAATY, T. L. (1980). *The Analytic Hierarchy Process.* New York: McGraw–Hill.

SHACHTER, R. D. (1985). Intelligent probabilistic inference. *Workshop on Probability and Uncertainty in Artificial Intelligence,* UCLA, 14–16 August 1985, RCA and AAAI.

SMITH, R. G. (1983). Strobe: support for structured object knowledge representation. *Proceedings International Joint Conference on Artificial Intelligence 1983,* pp. 855–858.

SMITH, R. G., DINITZ, R. & BARTH, P. (1986). Impulse-86: a substrate for object-oriented interface design. *Proceedings of the ACM Conference on Object Oriented Programming Systems, Languages, and Applications.*

SPETZLER, C. S., HOLSTEIN, C.-A. & VON STAEL, S. (1977). Probability encoding in decision analysis. *Readings in Decision Analysis,* Decision Analysis Group, SRI International, Menlo Park, CA.

ZIMMER, A. C. (1985). The estimation of subjective probabilities via categorical judgements of uncertainty. *Workshop on Probability and Uncertainty in Artificial Intelligence,* UCLA, 1985, 14–16 August, RCA and AAAI.