How To Do With Grids What People Say You Can't

Jeffrey M. Bradshaw, John H. Boose, Stanley P. Covington, and Peter J. Russo

Knowledge Systems Laboratory
Advanced Technology Center
Boeing Computer Services
P.O. Box 24346, M/S 7L-64
Seattle, Washington 98124 USA
(206) 865-3422
jbrad@Boeing.com

ABSTRACT

Although the representation of knowledge in grids is convenient for many purposes, there are still some significant limitations. One of the most challenging tasks still remaining is to evaluate the use of grids and personal construct methodology in problems that require constructive alternative generation and access to external tools and databases (i.e., *synthesis* as opposed to *analysis* problems). In this paper, we describe three components of an approach in support of this objective: 1. Definition of grid-based methods for interactive alternative generation and constraint discovery; 2. Development of an "open architecture" knowledge acquisition and decision support environment that supports asynchronous communication with external procedures and applications; and 3. Development of facilities for more sophisticated control and guidance of during knowledge acquisition and inference. These components are being implemented in *la folie à deux* (FAD), an environment that integrates components from *Aquinas,* a knowledge acquisition workbench based on personal construct theory, with those of *Axotl,* a knowledge-based decision analysis workbench.

1. INTRODUCTION

1.1. Strengths and Limitations of Grids for Knowledge Representation

Grids (Kelly, 1955) have many advantages as a general purpose form of knowledge representation. They may be viewed as a component of a database in entity-attribute form (Chen 1980): a grid has elements as *entities,* constructs as *attributes,* and allocations of elements to locations on construct dimensions as *values*.

The organization and logic of expert knowledge in a grid can be easily inspected and analyzed. Recognition and completion of patterns in the data are facilitated by the structure and relative compactness of the matrix representation as compared to rules. Furthermore, representation in grids facilitates testing for conditions of ambiguity, redundancy, and completeness (Cragun & Steudel, 1987).

Although a grid representation is convenient for many purposes, some significant limitations still exist. In an early paper describing the Expertise Transfer System (ETS)[1], Boose (1985) outlined four fundamental things that were difficult to do with single grids:

[1] ETS is an automated knowledge acquisition tool based on personal construct theory that was developed at Boeing Computer Services (Boose, 1986). It is a forerunner of *Aquinas*, a much more powerful tool that extends the capability in ETS (Boose & Bradshaw, 1987).

Abstraction levels. Attempts to represent knowledge at varying levels of abstraction created problems in ETS. For instance, in a maintenance system, a person might attempt to include the elements “engine” and “ignition coil” in the same grid. This caused difficulties when using grid elicitation techniques and some analysis tools.

Constructs that are not bi-polar. Typically, dimensions are represented in a grid as bi-polar constructs (e.g., a construct named “temperature” might have the poles “cold” and “hot”). In some situations (e.g., a series of constructs having to do with computer types (VAX / NOT-VAX, IBM / NOT-IBM, and so on)), it would be much easier to combine several constructs into a single nominally-scaled (“multi-polar”) construct (COMPUTER-TYPE with values are VAX, IBM, and so on).

Procedural and strategic knowledge. Eliciting and using deep *causal, procedural,* or *strategic* knowledge was difficult. Personal construct techniques were originally developed to describe entities rather than processes.

Synthesis problems. Handling *synthesis* problems such as design and planning, where the main task is *constructing* feasible alternatives rather than selecting from a pre-enumerated set (e. g., as in simple classification or diagnosis) was generally not practical for grid-based systems. Constraint-based reasoning was not available. Even if the large numbers of potential design alternatives could be pre-enumerated, a single grid representation would prove unwieldy and difficult to understand.

1.2. Progress in Grid Representation in *Aquinas*

It is heartening to look back over the last few years and recognize that, in cooperation with our colleagues in knowledge acquisition research, we have been able to make some progress on each of these issues (see Boose & Bradshaw, 1987; Boose, Shema & Bradshaw, 1988):

Abstraction levels. In our work on the *Aquinas* knowledge acquisition workbench, we have allowed individuals to have both grid and structured network views of the same information. The network views are a convenient way of representing different kinds of abstraction relationships (cases, experts, elements), and information and preference hierarchies (constructs). In *Aquinas,* for example, “engine” and “ignition coil” could be represented within two linked grids at different levels of abstraction.

Constructs that are not bi-polar. We have extended the representation of constructs to allow for unordered values. For example, an expert can have a construct COMPUTER-TYPE with values are VAX, IBM, and so on, or COLOR with values of red, green, and blue. We have also allowed the use of interval and ratio-scaled constructs and the representation of distributions of values as ratings (rather than just a single point value) within a particular grid cell.

Procedural and strategic knowledge. Some limited experimental work was performed to evaluate the use of grids for representing strategic and procedural knowledge as we designed the Dialog Manager (Kitto and Boose, 1987) and during development of a medical aid advisor (Boose, Shema & Bradshaw, 1988). Elicitation of medical procedures was accomplished by asking experts to compare and contrast different kinds of emergency situations.

Synthesis problems. Of the four things that were difficult to do with grids, we have put the least effort to date into this one. We would like to be able to evaluate the use of grids and personal construct methodology in problems that require constructive alternative generation and access to external tools and databases. In this paper, we describe some preliminary components of our approach.

1.3. Overview of the Modeling Process for Analysis and Synthesis Problems

There is a traditional distinction in the literature between *analysis* and *synthesis* problems (Rubinstein, 1975; Wise, 1985). Analysis problems are those in which the alternatives can be conveniently enumerated (e.g., classification, diagnosis, prediction), while synthesis problems are those where the main task is *constructing* feasible alternatives in a manner that is consistent with hard constraints and optimal (or "good enough") with respect to objectives ("soft constraints") (e.g., design, planning, configuration, scheduling). As Clancey (1984) observes, real-world problems do not always fall neatly into one of these two categories:

> "For example, if it were practical to enumerate all of the computer configurations R1 [an expert system that configures VAX computers] might select, or if the solutions were restricted to a predetermined set of designs, the program could be reconfigured to solve its problem by classification.
>
> Furthermore, as illustrated by ABEL [an expert system for medical diagnosis], it is incorrect to say that medical diagnosis is a 'classification' problem. Only routine medical diagnosis problems can be solved by classification… When there are multiple, interacting diseases, there are too many possible combinations for the problem solver to have considered them all before."

Our approach assumes that both analysis and synthesis techniques are relevant to the solution of most complex problems.

We can represent a typical approach to an analysis problem in terms of a three stage closed-loop diagram (Figure 1; adapted from Holtzman, 1989). During the *formulation* stage, an initial model is developed. *Evaluation* is the process of obtaining a formal recommendation by subjecting the model to some algorithm or inference procedure. The *appraisal* stage subjects the model to scrutiny to help the decision maker determine if there is sufficient basis to act, or if additional refinement of the model is needed[2].

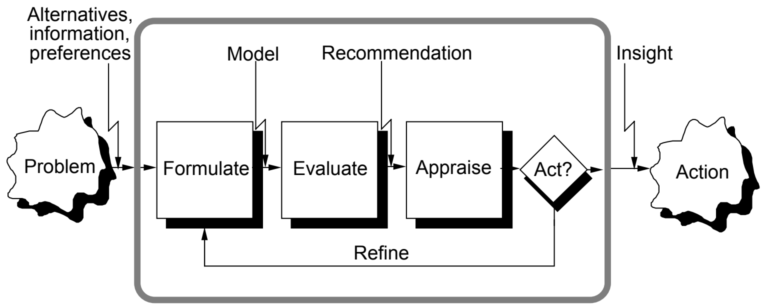

Figure 1. A typical approach to an analysis problem including formulation, evaluation, and appraisal stages, and a refinement loop.

A number of approaches have evolved in the literature to the solution of synthesis problems. More traditional approaches from fields such as operations research include optimization techniques (e.g., linear, non-linear, and dynamic programming) and the modeling of dynamic systems (e.g., control theory, simulation). Knowledge-based approaches have largely bypassed these methods in favor of various heuristic constraint satisfaction techniques. These approaches sometimes employ a variation of an incremental "propose-and-revise" strategy with the aim of *satisficing* rather than *optimizing* a solution (Marcus, 1988).

Traditional and knowledge-based approaches are similar in many respects. Both usually begin with a statement of desired outcome states in the form of constraints and attempt to synthesize *feasible* or *acceptable* alternatives through exploration of the bounds of the constraints. Since there will likely be several acceptable alternatives, the synthesis phase will be followed by an analysis to determine the best alternative with respect to a set of objectives such as least cost, maximum reliability, and so forth. Approaches to such problems generally do not differ with respect to these general steps, but rather in what specific methods are applied to generate hypotheses and reduce the extent of the search for a good solution. Figure 2 shows a typical approach to synthesis problems[3].

[2] Obviously, thinking of problem solving in terms of these stages is somewhat artificial.

[3] This figure is also a distortion of what really happens in problem solution — analysis and synthesis are both present in all phases of the solution process. Often it is more convenient to elicit constraints before generating alternatives.

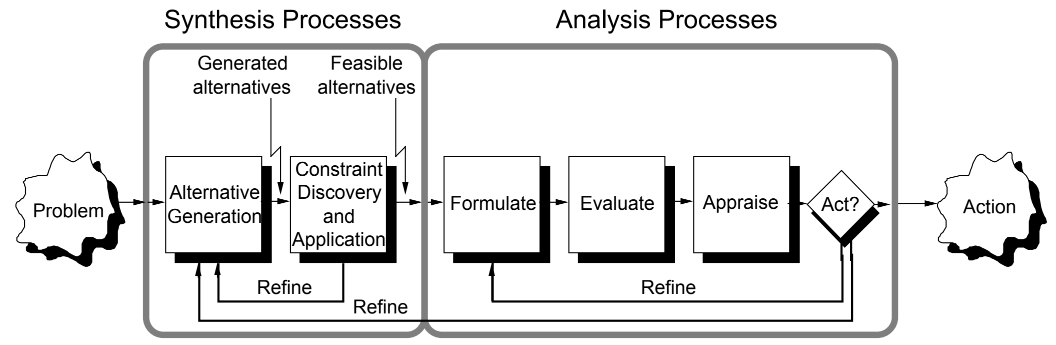

Figure 2. A typical approach to a synthesis problem adding alternative generation, and constraint discovery and application.

In order to support the use of grids and personal construct methodology as part of an approach to synthesis problems, we are pursuing the following objectives:

1. Define grid-based methods for interactive alternative generation, and constraint discovery and application. The effective generation of alternatives is crucial to any type of problem, but especially so in synthesis domains where one must deal with a combinatorial explosion of possibilities. Even the most sophisticated approach to the comparison of alternatives will produce unsatisfactory results if the universe of options being selected from is insufficiently rich and diverse. We are implementing a *possibility table* facility to aid in interactive alternative generation along with *constraint tools* that will assist in the discovery of constraints and in their application in limiting the solution space.

2. Development of an "open architecture" knowledge acquisition and decision support environment that supports asynchronous communication with external procedures and applications. Access to external procedures, applications, and data is crucial to any system that must deal with the complexity of real-world applications. Synthesis problems typically demand a higher degree of custom programming to accommodate the special needs of the domain than do analysis problems. We have designed and implemented an inter-application communication manager called *MANIAC* that supports the integration of *Aquinas, Axotl,* and other applications such as spreadsheets, databases, and hypermedia tools.

3. Develop facilities for more sophisticated control and guidance during knowledge acquisition. Coordination of tasks between multiple applications requires a level of control sophistication that is far beyond what is offered in the current version of *Aquinas.* We have developed a suite of knowledge-based tools to guide persons in their interaction with the system, thus minimizing the need for specialized training. The most important of the knowledge-based tools are the *activity graph facility,* the *agenda manager,* the *rule-based inference engine,* the *status board facility,* the *heuristic advisor,* and the *process executive.*

We are implementing these ideas within the *FAD* environment which is under development at Boeing Computer Services.

2. APPROACH

2.1. FAD — An environment to support *Aquinas* and *Axotl*

As a means of evaluating our concepts for an open architecture version of *Aquinas* for synthesis problems, we wanted to select another knowledge-based system being developed in our lab that could serve as the first major target for integration. We concluded that the *Axotl* knowledge-based decision analysis workbench would be well suited for the attempt from the standpoint of software compatibility and because we saw the two approaches as essentially complementary. Descriptions of *Aquinas* and *Axotl* are given below in section 2.3.

Preliminary work on the problem of combining *Aquinas* and *Axotl* functionality began in 1987, when both systems were being developed on the Xerox family of workstations in the Interlisp-D environment (Bradshaw & Boose, 1988). This work had been temporarily delayed because of changes in development platforms, however we recently succeeded in getting versions of both programs, *MacQuinas* and *MacXotl*, up and running on the Apple Macintosh. With both programs available within a single platform, we could begin work on the FAD environment. When completed, FAD will integrate components from *Aquinas,* a knowledge acquisition workbench based on personal construct theory, with those of *Axotl,* a knowledge-based decision analysis workbench.

In Bradshaw & Boose (1988), we outlined a rationale and preliminary approach to combining decision analysis with personal construct methodology for knowledge acquisition and problem solving. Decision analysis (Howard, 1966; Howard and Matheson, 1984; Keeney & Raiffa, 1976; Raiffa, 1968; Von Winterfeldt & Edwards, 1986) has been used for many years as a way to gain insight regarding decisions that involve significant amounts of uncertain information and complex preference issues. However decision analysis has been largely overlooked by knowledge-based system researchers who have generally opted for ad hoc, heuristic approaches to decision making and problem solving. These attempts to find heuristic alternatives to formal analysis have proven successful for many domains; however there have been concerns raised about the effectiveness of knowledge-based approaches for complex problems. Recently, several researchers have begun to look at decision analysis representations such as influence diagrams as a replacement for heuristic rule or frame-based approaches (see e.g., Horvitz, Breese & Henrion, 1988; Moore & Agogino, 1987; Rege & Agogino, 1988).

Meanwhile, practitioners of decision analysis have found that the greatest barriers to the acceptance of the methodology is that formal decision models may be too difficult and time-consuming for the typical decision maker to effectively build, use, and understand. As a result, some decision analysis practitioners are turning to knowledge-based system approaches in an effort to assist non-specialists in the development and evaluation of decision analysis models (e.g., Holtzman, 1989; Keeney, 1986; Wellman, 1986).

We are convinced that additional research will further demonstrate the complementary nature of decision analytic and knowledge-based systems approaches. The development of FAD will allow us to assess the value of combining these two methodologies within a single software environment (Figure 3). MANIAC (4; MANager for Inter-Application Communication) provides support for communication between *MacQuinas* (2), *MacXotl* (3) and other applications (5) throughout the modeling process. Alternative generation and constraint tools (1) and knowledge-based tools for control and guidance (6) are implemented within *MacXotl* to extend the functionality of the tools for more complex problems. Each of these components will be discussed in greater detail in sections 2.2-2.4.

FAD is an acronym derived from a peculiar interpersonal pathology called *la folie à deux* (loosely translated as "double madness"). Originating in a famous study by two French psychiatrists over a hundred years ago (Lasègue & Falret, 1877), the term has come to possess several shades of meanings:

> 1. "The presence of the same or similar delusional ideas in two persons closely associated with one another." (Gove, 1986);
>
> 2. "the influence of the insane on the supposedly sane man [and] the influence of the rational on the deluded one [;] through mutual compromises the differences are eliminated." (Lasègue & Falret, 1877);
>
> 3. The special relationship formed between two persons who are individually inadequate but who, as a couple, manage to sustain an acceptable social façade (Watzlawick & Weakland, 1977; Watzlawick, Weakland, & Fisch, 1974)[4,5].

[4] Watzlawick, Weakland, & Fisch (1974) cite related work by Lidz (1958) on the *transmission of irrationality,* Wynne's concept of *pseudo-mutuality* (1958), Laing's *collusion* (1969) and *mystification* (1965), Scheflen's *gruesome twosome* (1960), and Ferreira's *family myths* (n.d.).

[5] One reviewer has heard the term used in the opposite sense, namely to refer to two individually reasonable people who indulge in craziness when operating in tandem. Perhaps this interpretation will apply as well.

All shades of meaning are applicable here. First, if it is madness, as some claim, to model human problem solving on a computer using decision analysis or knowledge-based system methodology, it is certainly “double madness” to attempt to combine them. Secondly, some decision analysts regard the use of knowledge-based systems approaches for complex decision making as irrational; knowledge-based systems researchers typically feel that it is folly to model human beings rationally; since it is so difficult to determine who is whom, we will work toward “mutual compromises” to eliminate the differences. Thirdly, not only is this special relationship between a decision analytic approach and a knowledge-based one desirable, it is completely necessary in order to maintain the proper façade of respectability — either approach without the other, we think, leads to its own special brand of pathology.

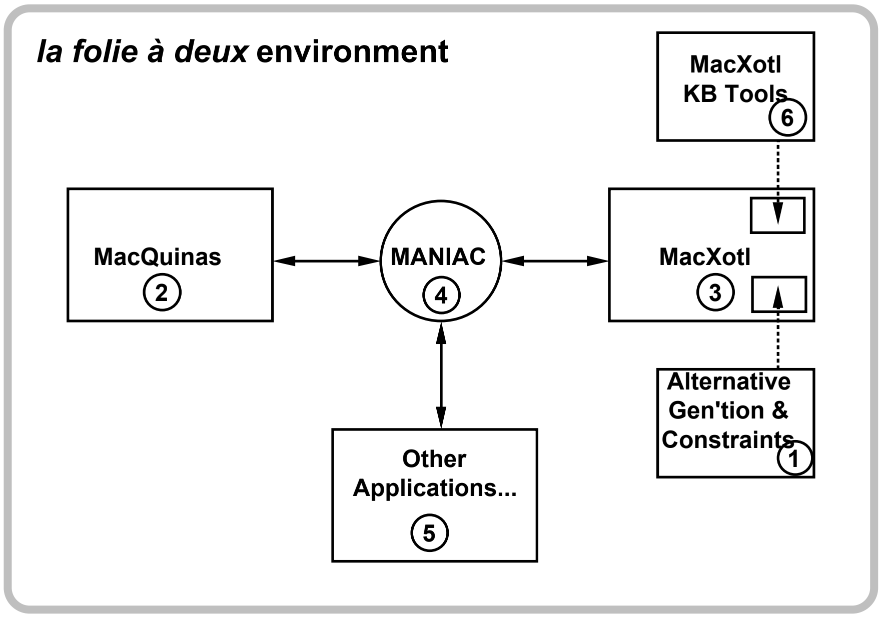

Figure 3. The *folie à deux* environment integrates components from *Aquinas*, a knowledge acquisition workbench based on personal construct theory, with those of *Axotl*, a knowledge-based decision analysis workbench.

Looking a little more closely at each of the stages, we can get a basic understanding of how *Aquinas, Axotl*, and applications such as databases and spreadsheets work together throughout the modeling process (Figure 4). The alternative generation and constraint tools will be implemented in Smalltalk-80 as part of the *MacXotl* workbench. The formulation stage depends on interviewing tools within both *MacXotl* and *MacQuinas.* We anticipate that the methodology from personal construct theory will be especially useful during this stage, particularly when the system is dealing with issues for which little domain knowledge is available. Prior to evaluation, grid and hierarchy structures in *MacQuinas* are converted to a representation compatible with influence diagrams, so that the evaluation algorithms within *MacXotl* can be used. External procedures and applications may also be called during evaluation as needed. Appraisal will rely on analysis tools residing in both *MacQuinas* and *MacXotl.*

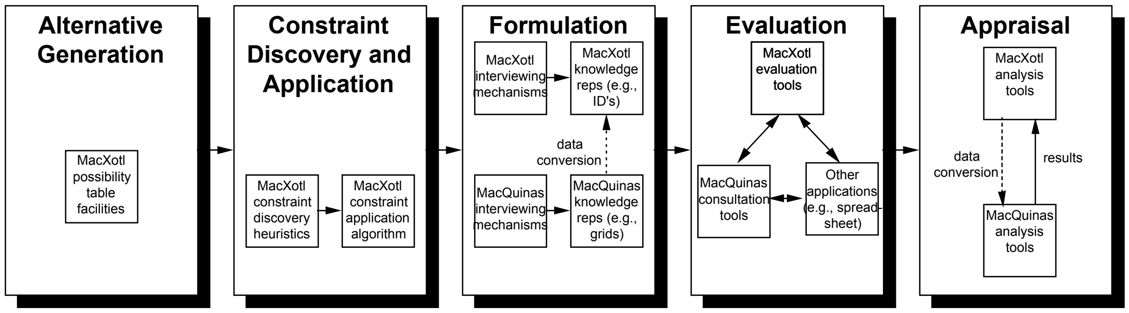
Alternative Generation
MacXotl possibility table facilities
Constraint Discovery and Application
MacXotl constraint discovery heuristics
MacXotl constraint application algorithm
Formulation
MacXotl interviewing mechanisms
MacXotl knowledge reps (e.g., ID's)
data conversion
MacQuinas interviewing mechanisms
MacQuinas knowledge reps (e.g., grids)
Evaluation
MacXotl evaluation tools
MacQuinas consultation tools
Other applications (e.g., spread-sheet)
Appraisal
MacXotl analysis tools
data conversion
results
MacQuinas analysis tools

Figure 4. Use of FAD components within an approach to synthesis problems.

We believe that grids are an advantageous form of knowledge representation for several modeling tasks. When speaking of grids, we do not restrict ourselves to the usual kinds possessing elements as columns and constructs as rows, but also include other grid-like matrix representations used to highlight relationships between different sorts of row and column variables.[6] Specifically, we will introduce *possibility tables* and *interaction grids* as representations that can be useful as persons generate alternatives and specify constraint relationships (sections 2.2 and 3 below).

While other representations are more useful for some purposes, there are often unique advantages to using grids. Jones (1981) explains how grid-like diagrams and network representations can complement one another:

> "Nets, point graphs, block diagrams, flow diagrams, circuit diagrams and the like are all applications of the convention of representing connections by a pattern of lines. The only advantage of a net over a matrix is the ease with which net patterns can be perceived and the problem understood. Matrices and nets are complementary ways of expressing a single set of relationships. The matrix enables a pattern that is too complex for the brain to generate all at once to be built up piece-by-piece outside the brain. A net of the same connections permits the assimilation of this pattern, once it has been completed and checked, back into the brain from whence came its constituent parts. Thus the brain can use an external aid to discover patterns among pieces of information that were originally understood only in isolation. Patterns get too difficult to perceive *as a whole* if there are more than fifteen to twenty elements; large networks are seldom any use as problem clarifiers."[7]

2.2. Constructive Alternative Generation and Constraint Tools

Issues in alternative generation. While research has led to greater understanding of techniques for *evaluating* competing alternatives (e.g., systems engineering techniques to estimate system reliability and maintainability, constraint propagation and multi-attribute utility approaches), relatively less effort has been focused on the problem of alternative *generation*. Although it would be impossible in practice to guarantee that all relevant alternatives were indeed identified, the effective generation of alternatives is crucial to the design process. Even the most sophisticated approach to the comparison of alternatives will produce unsatisfactory results if the universe of options being selected from is insufficiently rich and diverse.

Cognitive scientists have long known that humans typically retrieve only a small fraction of available alternatives in hypothesis generation tasks (Wise, 1985). Furthermore, persons tend to anchor on initial guesses, giving insufficient regard to subsequent data. For various other reasons, people may not be able to visualize whole classes of possibilities (Kahneman, Slovic, & Tversky, 1982).

At least two issues are important when approaching the task of alternative generation for synthesis problems such as design: (a) how can one determine if the designer has adequately considered the relevant and feasible alternatives; and, (b) which phase of the program life cycle (e.g., proposal preparation, concept definition, production) is most appropriate for generating and structuring the alternatives. For highly complex systems, significant penalties in cost and performance could result from postponing the decision to select a final design to the later stages of the program life cycle. Yet quite often, it may not be possible to accurately estimate system reliability and maintainability until the system approaches the last few stages of program life cycle. Techniques that can aid in capturing the design alternatives before such penalties can become significant or irreversible need to be investigated.

---

[6] This inclusive use of the term *grid* is not without precedent in the personal construct literature. Over the years a number of alternative grid forms have evolved that differ from the original repertory grid concept (e.g., implication grids, resistance-to-change grids, bi-polar implication grids, dependency grids, exchange grids, mode grids; see Fransella & Bannister, 1977; Shaw, 1981).

[7] Unfortunately, some have thought that personal construct methods could only be used effectively in conjunction with knowledge represented in grids. Within *Aquinas,* knowledge is stored as a network from which grids are dynamically constructed. The same information can be displayed and used in many different forms. George Kelly himself came to regret the narrow-mindedness of researchers who had equated personal construct theory and grids. Ten years after his initial two-volume work was published, he told Hinkle (1970) that if he were to revise the work he would probably delete the section on the repertory grid, because it seemed to him that "methodologically-oriented researchers had let it obscure the contribution of the theory."

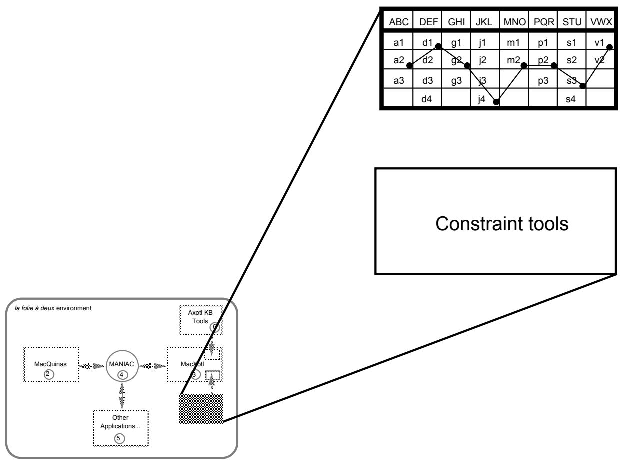

Figure 5. A possibility table facility working closely in conjunction with constraint tools provides an interactive interface for alternative generation, constraint discovery, and constraint application tasks.

The use of knowledge-based systems has been proposed both as a way to preserve a more complete record of available alternatives (i.e., capturing knowledge), and to help persons explore feasible combinations of design parameters that might otherwise go unconsidered. To this end, we are developing knowledge-based constraint tools that can work in conjunction with automated alternative generation facilities (Figure 5).

Possibility tables and constraint tools. Manually developed "strategy tables" have been used for many years by decision analysts as one way of generating and representing complex alternatives (McNamee & Celona, 1987). Related approaches, such as Zwicky's (1969) "morphological charts", have been manually employed by designers for many years. We plan to automate this representation and to extend its logic and structure to be more applicable to the types of problems we expect to encounter (e.g., hierarchical tables, explicit representation of constraints and preferences; Covington, 1987). We call this extended, automated representation a *possibility table.*

Possibility tables are a convenient way of structuring information relating to complex alternatives, outcomes, or plans[8]. Figure 6 shows portions of a possibility table that was developed manually at one point during *Axotl* development when we were determining implementation priorities. At that time, we were facing a tight deadline for a demonstration to management of the system that would show off some of the more innovative features. We also had a a requirement that our system meet a level of functionality required by users. There were also some decisions to make about the amount of resources that we would apply to the project. We knew we would not have enough time to do everything.

[8] Possibility table is the generic term referring to the graphical representation shown in Figures 6 and 17. We sometimes use the more specific terms *strategy tables* to refer to tables defining alternatives, *outcome tables* to refer to tables defining outcomes, and *agenda tables* to refer to tables defining portions of a plan.

POSSIBILITIES		COMPONENTS																	
		Spreadsheet		Home-Brew Prolog		Document'n of Code		Probability Encoder		DA Knowledge Base		Journal and Transcript		Dialog Boxes		UNDO Facility		Tutorial and User Manual	
Tight Resources		None	0	Use XQP only	0	No add'l document'n	0	Leave as is	0	None	0	None	0	Sequential menus only	0	**None**	**0**	None	0
Base Case	27	Modify Modify existing spreadsheet	5	Straight port of IE w/o XQP emulation	2	**Cursory update of PD document**	**3**	**Complete wheels and bar charts**	**1**	**Single, skeletal KB**	**5**	Journal only	1	**Ad hoc dialog boxes (no generality)**	**2**	Support only a few commands	1	**Terse, thorough primer**	**3**
Flashy Demo		**Enhance existing spreadsheet**	**9**	**IE with XQP syntax emulation**	**3**	Throrough update of PD document	5	Add cumulative display	2	Single, saleable KB	12	**Basic transcript only**	**1**	Ad hoc but flexible dialog boxes	3	Support most commands	3	Verbose primer	5
		Include some fancy features	15	IE with XQP & optimization	5			Add discretization	4	Two saleable KB's	15	Journal and fancy transcript	5	Generic dialog box facility	7	Multiple UNDO's up to a definable threshold	5	Full-fledged manual with tutorial	10
All-out		Comm'l qual spreadsheet	40							Fancy, useable KB	35								

Figure 6. Possibility table for *Axotl* development strategy.

We decided which were the important system components that were to be implemented and placed names for them at the top of each column. Within each column, we place the various possibilities for levels of implementation for that system component. Next to each item, a number was placed representing the resources, in person-weeks, that we estimated would be necessary to accomplish that item.

Following this, we discussed the various objectives we were trying to accomplish during the next phase of development and imagined rationales for forming strategies that would emphasize or de-emphasize particular objectives (e.g., flashy demo vs. tight resources). Each strategy was given a name or theme and placed in the leftmost column.

For each theme, a "path" through the table is defined by selecting one item in each column in a manner that is consistent with the theme and compatible with the items previously selected and with global constraints (e.g., total person-weeks available). In this case, each path represents a different alternative for system development. These alternatives could then be used in turn within a decision node of an influence diagram or as elements in a grid during further refinement of the model.

Automating possibility tables would greatly enhance their usefulness. It would be possible to implement a top-down refinement strategy based on hierarchical possibility tables. General heuristics and domain specific help could be available to individuals during the formulation of possibilities. For large problems, it would be impractical to force persons to assign each item in a path directly. Where hard and soft constraints (preferences) were associated with items in the table, they could be used by the system to guide and constrain the definition of a path in a mixed-initiative mode. For instance, particular items would become grayed out if incompatible; other items could be pre-selected by default consistent with constraints, preferences, and previous choices in the table. For some types of problems, one could envision that the items in the table would be generated programmatically via a database lookup or some other kind of user-defined procedure.

Additional methods available in an automated environment will be used to help persons discover and apply attributes, components, and constraints. Psychological techniques derived from personal construct theory can help identify, structure, and refine discriminating characteristics of individual items and possibility paths, and to group similar ones. An iterative search procedure will be implemented which hypothesizes new constraints based on examples of previously-defined paths, and proposes new paths based on permutations of the constraint space. An example of the use of possibility tables for a design problem is given in section 3 below.

2.3. Open Architecture

Access to external procedures, applications, and data is crucial to any system that must deal with the complexity of real-world applications. Synthesis problems typically demand a higher degree of custom programming to accommodate the special needs of the domain than do analysis problems. We are developing FAD in such a way that it can be used as an integrating architecture for *Aquinas* and *Axotl.* We are also defining facilities so that other applications can be incorporated into a problem solving strategy when appropriate.

***Aquinas* — A knowledge acquisition workbench based on personal construct theory.** *Aquinas*, an expanded version of the Expertise Transfer System (ETS; Boose, 1986), is a workbench that supports several knowledge acquisition activities (Figure 7). The workbench combines techniques from psychology and knowledge engineering. Activities supported by *Aquinas* include eliciting distinctions, decomposing problems, combining uncertain information, incrementally testing knowledge bases, integrating data types, automatically expanding and refining the knowledge base, using multiple sources of knowledge, use of constraints in the inference process, and providing guidance during the knowledge acquisition process. *Aquinas* interviews experts and helps them analyze, test, and refine their knowledge base. Expertise from multiple experts or other knowledge sources can be represented and used separately or combined. Results from consultations are derived from information propagated through hierarchies. Distinctions captured in grids can be used directly in problem solving or converted to other representations such as production rules, fuzzy sets, or networks. *Aquinas* can deliver knowledge by creating knowledge bases for several different expert system shells or through internal consultation facilities. Help can be given to the expert by a Dialog Manager that embodies knowledge acquisition heuristics (Kitto & Boose, 1987).

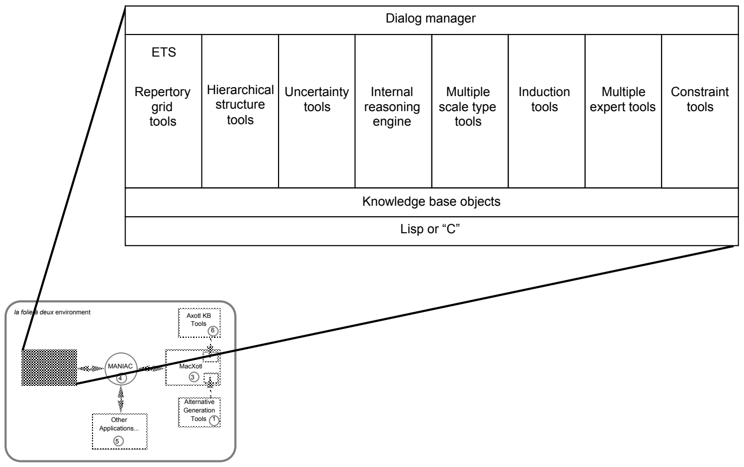

Figure 7. *Aquinas* is a knowledge acquisition workbench that combines a number of useful analytic tools.

Using *Aquinas*, small rapid prototypes of a knowledge-based system can be built in as little as one hour, even when the expert has little understanding of knowledge-based systems or has no prior training in the use of the tool. *Aquinas* uses methods from personal construct theory, an approach that grew out of George Kelly's research and experience as a clinical psychologist (Kelly, 1955). Kelly's methods and theory provide a rich framework for modeling the qualitative and quantitative distinctions that form an important part of part of an expert's problem-solving knowledge.

Aquinas is written in Interlisp and runs on the Xerox family of workstations and on the Sun in the Envos environment. Subsets of *Aquinas* also run in an Interlisp version on the DEC Vax. A C-based version has been ported to the Apple Macintosh (*MacQuinas*) and to a variety of UNIX platforms. *Aquinas* is discussed in greater detail in Boose & Bradshaw (1987), Boose, Shema & Bradshaw (1988), and Boose (1988).

***Axotl* — A knowledge-based decision analysis workbench.** *Axotl* integrates a set of computer-based decision analysis tools with a knowledge-based system (Figure 8). The decision analysis tools are designed for problems involving large amounts of uncertainty and complex tradeoffs. Decisions of this type are often ill-served by the heuristic inference mechanisms in conventional AI software. Fully-integrated knowledge-based tools use application-independent and application-specific knowledge from experts to guide users through the creation, evaluation, and appraisal of a formal decision model. The goal is to ensure that persons using the system will receive the same kind of assistance they would get if they were aided by a professional decision analyst.

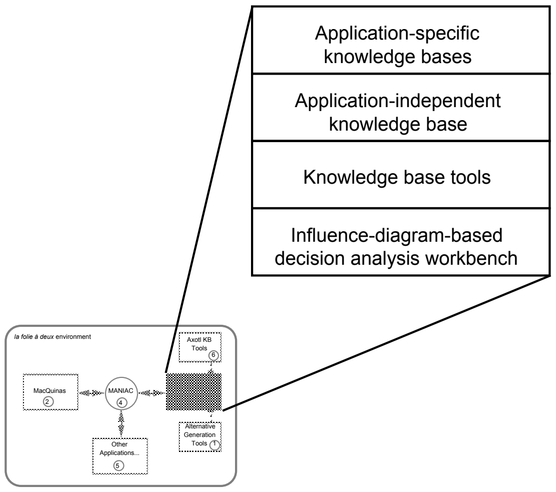
Application-specific knowledge bases
Application-independent knowledge base
Knowledge base tools
Influence-diagram-based decision analysis workbench
la folie à deux environment
Axotl KB Tools 6
MacQuinas 2
MANIAC 4
Other Applications... 5
Alternative Generation Tools 1

Figure 8. *Axotl* is a knowledge-based decision analysis workbench

A complete decision model, containing relevant items of problem-solving knowledge and their interrelationships, constitutes the *decision basis* (Howard and Matheson, 1984). Three things are represented in the decision basis: *information*, *preferences*, and *alternatives*. In a medical diagnosis and treatment decision, the information consists of the knowledge a physician possesses relating symptoms and diseases; the preferences consist of factors that determine the desirability of a treatment alternative, such as cost, effectiveness, or risk; and the alternatives consist of the various possibilities for treatment. These three types of knowledge and some important subtypes are shown in Figure 9.

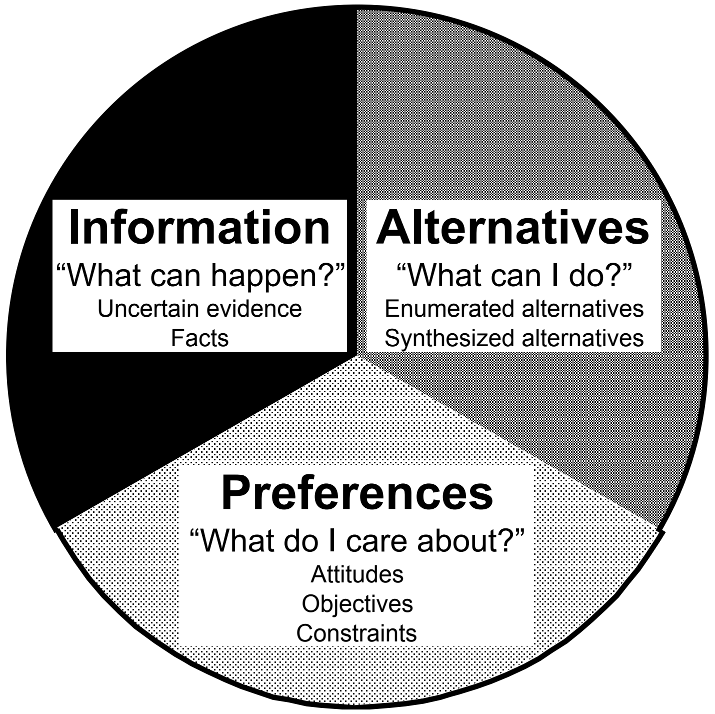

Figure 9. The decision basis is comprised of information, alternatives, preferences, and their interrelationships.

The influence diagram editor is the central tool of the decision analysis workbench. Influence diagrams describe information, alternatives, and preferences relevant to a decision both graphically and mathematically (Howard & Matheson, 1980). They can be directly solved to yield a recommended course of action and an expected value or utility for that action. While mathematically similar to probabilistic decision trees, they possess several advantages:

1. Influence diagrams grow linearly in their graphical representation as contrasted with the exponential growth of trees; 2. They can represent and exploit conditional independence; and 3. They can be integrated with external procedures and functions in a straightforward manner. Additionally, our experience confirms that influence diagrams are an effective way of communicating important issues among participants in a decision, even for those who may not understand the mathematical underpinnings.

Figure 10 shows a screen snapshot from *Axotl* of an influence diagram representing a fictional R&D investment decision problem. The problem is to determine an investment strategy for "Scribe", an automated speech-to-text transcriber, considering technical risks and market uncertainties. The investment strategy is composed of three decisions (development investment level, production investment level, and unit price) that are represented by rectangular nodes on the diagram. Oval nodes represent technical risk variables (accuracy, speed), production uncertainties (unit cost), and market uncertainties (market size). The eight-sided node labeled "Profit" has been designated as the criterion to maximize in evaluating the decision model to determine an optimal policy. Arrows between nodes represent relationships where influence or information is imparted from one variable to another. An additional type of node, not shown in this diagram, can represent a deterministic function. This allows the influence diagram to be transparently linked to external procedures or to programs such as databases and spreadsheets.

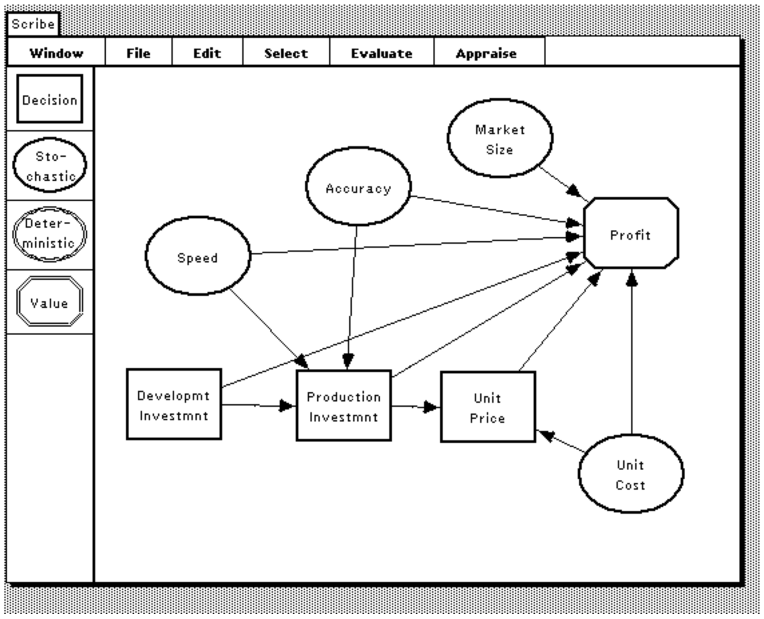

Figure 10. An influence diagram for an R&D investment decision about "Scribe", a fictitious automated speech-to-text transcriber

The *Axotl* influence diagram processing module is a general-purpose tool for editing influence diagrams and solving them to obtain recommended actions. As a result of the influence diagram solution process, an expected value or utility is computed for each alternative which which expresses the range of profit or loss that is anticipated for a given course of action. Often the greatest value in the use of the tool is not in obtaining an optimal recommendation, but in the insight gained from being able to pose and answer a variety of questions about decision variables (Bradshaw & Boose, 1988; Howard & Matheson, 1984). Sensitivity analysis enables individuals to determine which variables are the most important determinants of final value (e.g., "Is speed more important than accuracy?"). Value-of-information analysis is useful in understanding the importance of resolving uncertainty for specific components of the model (e.g., "How much should I spend to pin down the size of the market?"). Value-of-control analysis focuses attention on new alternatives that can increase our ability to bring critical uncertainties under our control (e.g., "Should we control unit cost by acquiring a semiconductor company that manufactures the required chips?").

Axotl's user interface provides a highly interactive graphic environment from which the person can operate the rest of the system. Decision analysis activity modules are oriented around specific tasks and computational activities that may be initiated by the person or invoked as a result of knowledge-based inference. Knowledge-based tools provide capabilities for the creation, use, and maintenance of decision analysis expertise stored in knowledge bases. The full functionality of the activity modules is available for both manual and knowledge-based control.

Application-independent knowledge bases contain expertise about specific tasks (e.g., probability encoding, risk assessment) and general knowledge about the process of decision analysis and the process of knowledge acquisition. Application-specific knowledge bases can transform *Axotl* from a general purpose decision analysis workbench to an application that is tailored for a specific class of decisions (see Figure 10). For example, we are currently building a prototype application named PIE (Project Investment Evaluation; Bertrand, Bradshaw, Covington & Holtzman, 1987; Bradshaw & Holtzman, 1987) that is being configured to assist management in making R&D project selection decisions. When finished, it will contain separate, compatible knowledge base modules for generic R&D decision-making knowledge, Boeing R&D knowledge, and project-specific knowledge.

Axotl was originally implemented on Xerox 1100-series artificial intelligence workstations, but is now being developed in ParcPlace Smalltalk-80 on the Apple Macintosh II (*MacXotl*). Smalltalk-80 is based on the concept of a virtual machine, whose virtual image runs unchanged on hardware platforms from various vendors. Versions of Smalltalk-80 exist for Sun, Apollo, Hewlett-Packard, IBM, and Apple hardware.

MANIAC: the inter-application communication manager. Figure 11 shows a high level view of message passing between applications within FAD. The key component to all these communication processes is "MANIAC" (MANager for Inter-Application Communication; Covington, 1988). MANIAC consists of two parts: an interface and a message manager.

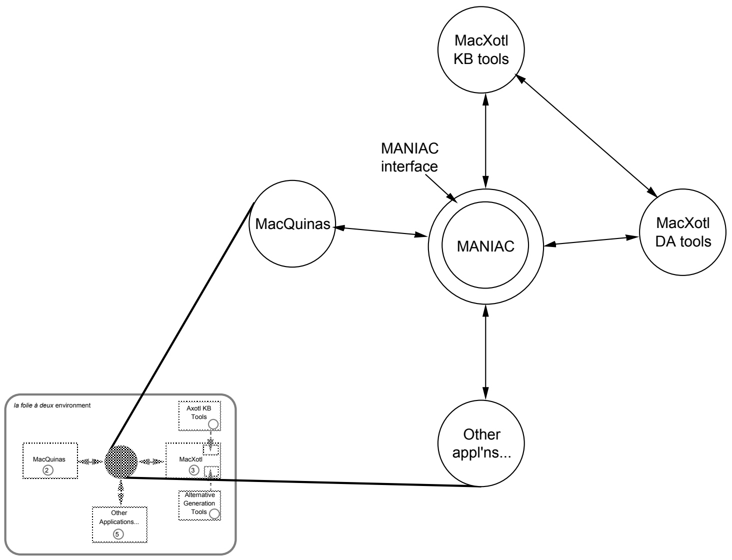

Figure 11. High-level view of message passing between FAD applications.

MANIAC supports asynchronous and synchronous communication between any number of applications. Each application need only support a single generic call to the interface[9]. The interface parses and routes messages to the manager, which maintains a message list. The message list is polled by potential receivers for messages, and by senders for replies.

[9] The MANIAC interface is implemented as an XCMD that is installed in the resource fork of each application (Bond, 1988). The MANIAC message manager is implemented as a device driver (Apple, 1985-1987). Applications such as Microsoft Excel, where we have no access to low-level programming hooks, must rely on automated command utilities such as Tempo II to provide some parts of the support necessary support for sending or receiving messages. However an increasing number of applications are beginning to include XCMD and/or HyperCard support. Applications that support XCMD calls will not only have access to MANIAC facilities, but also to the hundreds of general XCMD utilities that are available from vendors and in the public domain.

Using MANIAC, several applications may be launched and run concurrently in the foreground or background. The knowledge-based tools in *MacXotl* (described in below in section 2.4) do the major work of initiating and coordinating tasks between applications.

Other applications. Several MANIAC links between applications are in various stages of completion. Some of the more important ones are shown in Figure 12.

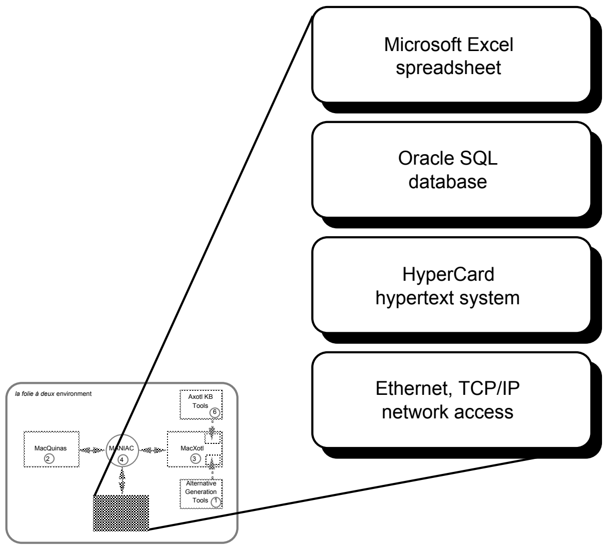
Microsoft Excel spreadsheet
Oracle SQL database
HyperCard hypertext system
Ethernet, TCP/IP network access
la folie à deux environment
Axotl KB Tools 6
MacQuinas 2
MANIAC 4
MacXotl 3
Alternative Generation Tools 1

Figure 12. Some of the applications that will be linked to the FAD environment via MANIAC.

The *MacQuinas/MacXotl* link permits the two applications to pass messages and data structures to one another, making the functionality of each tool available to the other. Since the interface of *MacQuinas* is character-based and hence lacks many of the useful graphical facilities in the Interlisp version, we are exploring the feasibility of creating a graphical interface to the C program in Smalltalk.

A rudimentary link to Microsoft Excel was previously implemented in the *MacXotl* environment to provide the capability of linking spreadsheet models to influence diagrams. We are re-implementing this link to take advantage of the additional features available in MANIAC.

Access to databases on stored on mainframes is very important for these tools to be effectively fielded within Boeing. Oracle provides support for the SQL queries of internal and external databases through a programmatic interface and also within the HyperCard environment.

HyperCard (Goodman, 1987) is an important component of FAD, since a growing number of applications are using it as a graphical front-end. It provides a common, extensible interface that can be easily customized by end users to create interactive tutorial and reference materials, or merely to change the appearance of the interface according to their preferences. We plan to build tools that allow access to high level "programming languages" such as HyperTalk for problem specific data, end user interfaces, and knowledge base tools in a way that shields domain experts and end users from having to learn Smalltalk as they configure and extend the environment.

An Ethernet link to MANIAC will provide access to external applications and procedures. Ideally, these external links would operate transparently to the person using the system, in the same fashion as the internal links do.

2.4. Sophisticated Control and Guidance During Knowledge Acquisition and Inference

Knowledge-based tools for control and guidance. In an effort to manage the increased complexity of *Aquinas* as the size of problems grew and the number of options available to the individual increased, Kitto and Boose (1987) developed a Dialog Manager facility. The Dialog Manager subsystem of *Aquinas* incorporates a set of knowledge acquisition heuristics to provide guidance to domain experts and knowledge engineers. While the facility has been successful in providing limited help to persons learning *Aquinas,* the relatively unstructured, rule-based format of the knowledge base has always been difficult to maintain and keep consistent with the changing *Aquinas* development environment. With the need for increased sophistication in control and guidance of a session, we knew we needed to try a more adequate representation for control knowledge. To this end, we are extending and adapting knowledge-based tools originally developed to manage sessions with *Axotl* to control and coordinate tasks that may span several applications.

When fully implemented, there will be six major components of the knowledge-based tool set: the *activity graph facility,* the *agenda manager,* the *rule-based inference engine,* the *status board facility,* the *heuristic advisor,* and the *process executive* (see Figure 13). We will describe each one of these in turn.

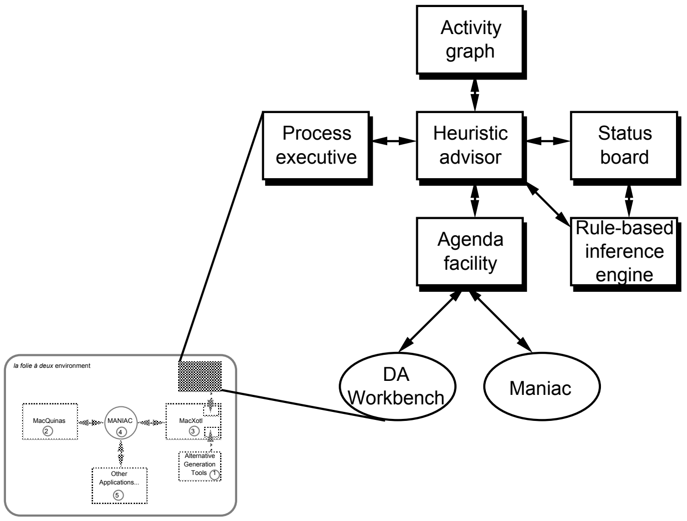

Figure 13. *Axotl* knowledge-based tool architecture.

1. Activity graph facility. An activity graph is a representation of the consultation process as a hierarchy of goals and activities (Bertrand, Bradshaw, Covington & Holtzman, 1987; Russo, 1988). The topmost goal in the hierarchy represents the successful completion of a consultation; subgoals and activities to support them are added and pruned from the hierarchy dynamically as a consultation with the system proceeds. Each goal in the hierarchy has an associated set of conditions that must either be satisfied by the completion of supporting activities or explicitly overridden by the individual. Figures 14 and 15 are portions of activity graphs that were developed as part of the PIE project to illustrate how automated help could be provided to persons making R&D project selection decisions.

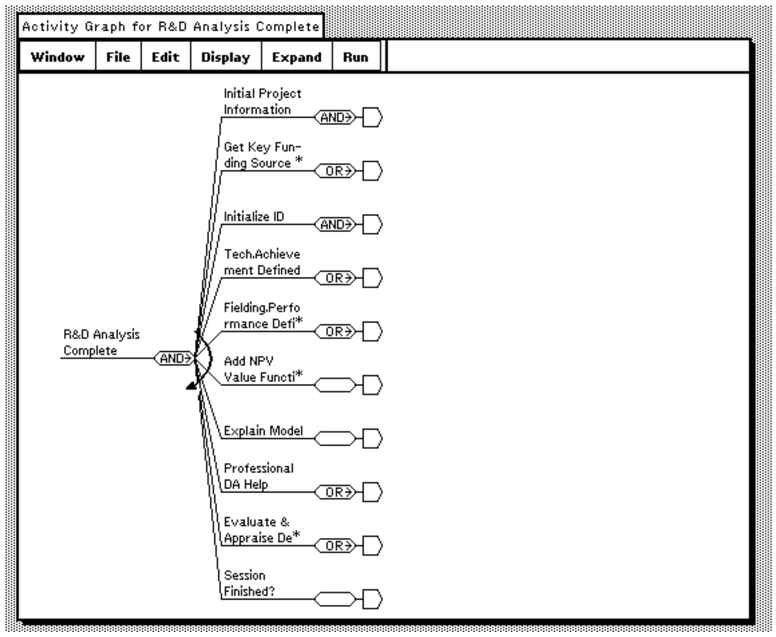

Figure 14. Portion of an activity graph containing goals to be satisfied for completion of a successful R&D project selection consultation. The ends of each branch are truncated so that only the topmost level is visible.

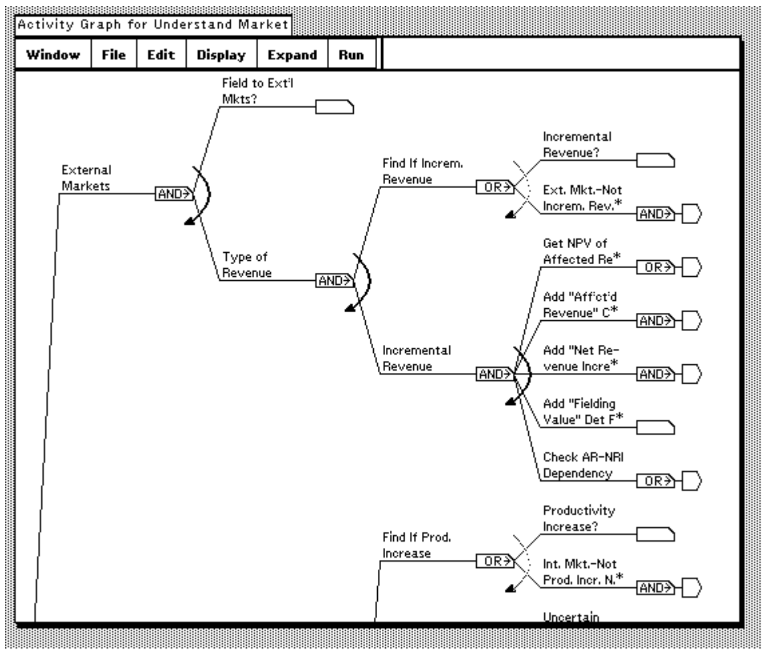

Figure 15. Some subgoals and activities in the R&D project selection activity graph.

Activity graphs are similar in some respects to AND/OR graphs familiar to knowledge-based system researchers, but contain features that are specialized to their use in this application (e.g., representation of iteration, directedness of conjunctions and disjunctions, differentiation between goals and activities, dynamic cost/benefit functions at nodes to assist in setting priorities for goals). In our design, goal definition and control are explicitly separated: the goal and activity components consist of strictly declarative activity graph pieces, while the procedural aspect of session control is allocated to the process executive and heuristic advisor (described below). Activity graphs are built up, pruned, and modified during consultations as a result of knowledge-based inference by the heuristic advisor, which continually monitors the progress of the session.

The activity graph facility includes components for viewing and editing goals and activities graphically. Persons can create and modify activity graph pieces through this graphical editor. During a consultation, the current status of goals and activities may be signalled graphically so that persons can monitor progress and manually override the normal execution sequence if they desire.

2. Agenda manager. The agenda manager maintains a structure that contains a list of activities that are sufficient to satisfy the consultation goals. An agenda may be thought of as a "cut set" through the activity graph. For a given activity graph, there may be several "cut sets" that could satisfy the goals — the agenda manager contains the one selected as best by the heuristic advisor.

The agenda is dynamic and changes many times during the course of a consultation. The agenda manager executes items on the agenda sequentially one by one until it is interrupted or modified in response to the failure of an activity or goal, or because of a some other change in conditions.

3. Rule-based inference engine. A full-featured rule-based engine was implemented in Smalltalk as a resource to the heuristic advisor. Its capabilities include forward & backward chaining, multiple worlds, and a graphical interface for defining predicates and formulating queries.

4. Status board facility. The status board facility allows the viewing and editing of the agenda and a variety of consultation status indicators. Status indicators are parameters used by the heuristic advisor as a part of setting task priorities. Indicators include information on consultation status, current user status, decision problem status, and decision model status.

5. Heuristic advisor. The heuristic advisor performs two functions, as called by the process executive:

A. Based on information from the status board, the heuristic advisor builds up, prunes, and modifies the current activity graph.

B. The heuristic advisor uses information in the knowledge base to generate the best possible analysis agenda. Functionally, it consists of a panel of "specialists", each of which evaluates possible activities with respect to a single criterion such as completeness, balance, precision and detail, and so forth. Based on the relative weights of the specialists at a given point in time and the strengths of their recommendations, a joint determination of the most desirable agenda is made. Personal construct techniques seem well-suited for acquiring portions of the knowledge base of these specialists (Boose, Shema, & Bradshaw, 1988).

The composition, priority, and recommendations of the specialists will change during the course of a consultation as a result of information in the knowledge base, the state of the decision model, and other status indicators contained in the status board.

6. Process executive. The function of the process executive is to schedule the actions of the heuristic advisor and the agenda manager, and to handle interrupts from the user and from external procedures and applications. The process executive repeatedly cycles through calls to the heuristic advisor to determine if the activity graph should be modified or if a new agenda should be generated and calls to the agenda to execute the next item.

3. EXAMPLE

To illustrate the use of the tools we have described, we will present a brief example of how one might go about designing a hand-held ink-writing instrument (portions adapted from Jones, 1981). There are five basic steps:

1. List functions that any acceptable design must be able to perform as column headers in a possibility table.
2. In columns of the possibility table, list a range of sub-solutions, i.e., alternative means of performing each function.
3. Explore constraint relationships within the possibility table.

4. Settle on a manageable number of sufficiently different alternatives by selecting sets of compatible sub-solutions.
5. Develop criteria for final choice, taking uncertainties into account.

Because of the focus of this paper on synthesis problems, we will focus on the first four of these steps.

1. List functions that any acceptable design must be able to perform as column headers in a possibility table. The major functions of a hand-held ink-writing instrument are assumed to be:

a. *Transfer.* How should ink be transferred to paper?

b. *Replenishment.* How should the ink reservoir be replenished?

c. *Protection.* How should the ink transfer mechanism be protected when not in use?

d. *Pocket position.* Which way should the pen be aligned in the pocket?

2. In columns of the possibility table, list a range of sub-solutions, i.e., alternative means of performing each function. Sub-solutions for each function are listed within the columns of the possibility table (see Figure 17). For example, a nib and a ballpoint are two ways to transfer ink to paper.

3. Explore constraint relationships within the possibility table. Two general types of constraints need to be considered: local constraints and global constraints. Global constraints include overall considerations such as cost, appearance, and reliability. Local constraints contain information about mutually incompatible or mutually necessary sets of sub-solutions, or synergistic or antagonistic relationships between them. The interaction grid shown in Figure 16 displays three incompatibilities between pairs of sub-solutions. The reasons for the incompatibilities are as follows:

a_1c_2 — Nibs require a sealed cover, but retracted transfer mechanisms are open to the air.
a_1d_2 — Nibs leak when they are stored point down in the pocket.
a_2b_1 — Ballpoint ink is too viscous to work with suction refill.

OPTIONS	a1	a2	b1	b2	c1	c2	d1	d2
Transfer	▨	▨						
a1 nib	▨	▨				I		I
a2 ballpoint	▨	▨	I					
Replenishment	▨	▨	▨	▨				
b1 suction refill	▨	▨	▨	▨				
b2 repl. reservoir	▨	▨	▨	▨				
Protection	▨	▨	▨	▨	▨	▨		
c1 replaceable cover	▨	▨	▨	▨	▨	▨		
c2 retractable point	▨	▨	▨	▨	▨	▨		
Pocket position	▨	▨	▨	▨	▨	▨	▨	▨
d1 point up	▨	▨	▨	▨	▨	▨	▨	▨
d2 point down	▨	▨	▨	▨	▨	▨	▨	▨

Figure 16. An interaction grid.

4. Settle on a manageable number of sufficiently different alternatives by selecting sets of compatible sub-solutions. In this small example, all possible sets can be enumerated:

$a_1b_1c_1d_1$	Conventional fountain pen
$\underline{a_1}b_1c_1\underline{d_2}$	Incompatible set
$\underline{a_1}b_1\underline{c_2}d_1$	” ”

$\underline{a_1}b_1\underline{c_2}d_2$	” ”
$a_1b_2c_1d_1$	Cartridge fountain pen
$\underline{a_1}b_2c_1\underline{d_2}$	Incompatible set
$\underline{a_1}b_2\underline{c_2}d_1$	” ”
$\underline{a_1}b_2c_2\underline{d_2}$	” ”
$\underline{a_2}\underline{b_1}c_1d_1$	” ”
$\underline{a_2}\underline{b_1}c_1d_2$	” ”
$\underline{a_2}\underline{b_1}c_2d_1$	” ”
$\underline{a_2}\underline{b_1}c_2d_2$	” ”
$a_2b_2c_1d_1$	Ballpoint with replaceable cover
$a_2b_2c_1d_2$	(A)
$a_2b_2c_2d_1$	(B)
$a_2b_2c_2d_2$	Retractable ballpoint pen

Constraints are so useful during this process because they can efficiently limit the space of effective search for new alternatives. Note that the three constraints disovered above serve to eliminate ten of the sixteen possible combinations of parameters.

The alternative generation/constraint discovery phase can continue indefinitely until the person is satisfied with the set of alternatives. Additional components of the possibility table can be elicited by asking questions about alternatives (“What is a functional component of regular fountain pens, cartridge fountain pens, and retractable ballpoint pens that makes two of them similar and different from the third?”). If the person has not exhaustively considered constraints, the system can generate new alternatives (e.g., A and B above) and ask whether they are feasible. If so, a novel alternative has been generated. If not, a new constraint has been discovered.

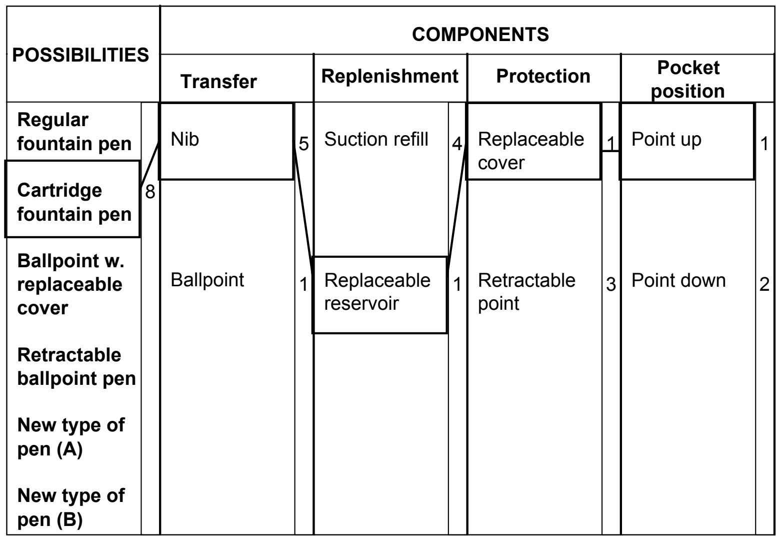

Figure 17. A possibility table containing design alternatives for a hand-held ink-writing instrument.

5. Develop criteria for final choice, taking uncertainties into account. Once the person is satisfied with the alternatives that have been developed, refinement of preferences can proceed. One way of eliciting preferences is to ask triad comparison questions again (“What is a desirable quality of regular fountain pens, cartridge fountain pens, and retractable ballpoint pens that makes two of them similar and different from the third?”). Information gathered in this way can be represented in an *Aquinas* grid and later transformed to an influence diagram within *Axotl.* Sensitivity analysis performed within *Axotl* focuses attention on the most critical variables where careful consideration of uncertainties is most warranted. Evaluation and appraisal tools assist in final refinement of the model and make recommendations about alternatives. Greater detail on *Aquinas* and *Axotl* formulation, evaluation, and appraisal methodology is given in Boose & Bradshaw (1987) and Bradshaw & Boose (1988).

4. CONCLUSIONS

Of the things we knew would be difficult to do with grids, providing support for constructive problem solving may well turn out to be the most challenging. For the present, we will be happy with even a modest initial success.

ACKNOWLEDGEMENTS

We express our appreciation to Kit Bakke, Miroslav Benda, Kathleen Bradshaw, Beverly Clark, Karolyn Covington, Cathy Kitto, Sandra Marcus, Art Nagai, D. "Carlos" Schuler, Kish Sharma, David Shema, Lisle Tinglof-Boose, and Bruce Wilson for their contributions and support. Samuel Holtzman, Ronald Howard, Keh Shiou Leu, James Matheson, Léonard Bertrand, and James Smith of Strategic Decisions Group were co-designers and co-developers of the original *Axotl* prototype. This work has also benefitted from ongoing discussions with Brian Gaines, Max Henrion, and Mildred Shaw. *Aquinas* and *Axotl* are under development at the Knowledge Systems Laboratory, Advanced Technology Center, Boeing Computer Services in Seattle, Washington.

REFERENCES

APPLE COMPUTER (1985-1987). *Inside Macintosh.* Volumes 1-5. Reading, Mass.: Addison-Wesley.

BERTRAND, L., BRADSHAW, J.M., COVINGTON, S., and HOLTZMAN, S. (1987). *PIE decision analysis workbench: Increment #2 preliminary design document.* Advanced Technology Center Technical Report BCS-G2511, November 1987. (controlled distribution)

BOND, G. (1988). *XCMD's for HyperCard.* Portland, Oregon: MIS Press.

BOOSE, J.H. (1985). A knowledge acquisition program for expert systems based on Personal Construct Psychology. *International Journal of Man-Machine Studies,* 23**,** 495-525.

BOOSE, J.H. (1986). *Expertise Transfer for Expert System Design.* New York: Elsevier.

BOOSE, J.H. (1988). Uses of repertory-grid-centered knowledge acquisition tools for knowledge-based systems. *International Journal of Man-Machine Studies*, in press.

BOOSE, J.H. & BRADSHAW, J.M. (1987). Expertise transfer and complex problems: Using *Aquinas* as a knowledge-acquisition workbench for knowledge-based systems. *International Journal of Man-Machine Studies,* 26**,** 3-28.

BOOSE, J.H., SHEMA, D.S. & BRADSHAW, J.M. (1988). Recent progress in Aquinas: A knowledge acquisition workbench. *Proceedings of the Third AAAI Workshop on Knowledge Acquisition for Knowledge-Based Systems*, Banff, Canada, November, 1988.

BRADSHAW, J. M. & HOLTZMAN, S. (1987). Artificial intelligence and decision analysis. Presentation made to the Nineteenth Annual Meeting of the Society for Information Management, Seattle: October 17, 1987.

BRADSHAW, J. M. & BOOSE, J.H. (1988). Decision analysis techniques for knowledge acquisition: Combining information and preference models using *Aquinas. International Journal of Man-Machine Studies,* in press.

CHEN, P.P., (ed.) (1980). *Entity-Relationship approach to Systems Analysis and Design.* New York: North Holland.

COVINGTON, S. C. (1987). Strategy table strategy. Internal technical memo.

COVINGTON, S. C. (1988). MANIAC Driver Specifications. Internal technical memo.

CRAGUN, B.J. & STEUDEL, H.J. (1987). A decision-table-based processor for checking completeness and consistency in rule-based expert systems. *International Journal of Man-Machine Studies,* 26, 633-648.

FERREIRA, A.J. (n.d.). Psychosis and the family myth. Unpublished manuscript.

FRANSELLA, F. & BANNISTER, D. (1977). *A Manual for Repertory Grid Technique.* London: Academic Press.

GAINES, B. R. (1988). Second-generation knowledge acquisition systems. in *Proceedings of the Second European Knowledge Acquisition Workshop.* Bonn, West Germany, June 19-23, 1988.

GOODMAN, D. (1987). *The Complete HyperCard Handbook.* New York: Bantam Books.

GOVE, P.B. (1986). *Webster's Third New International Dictionary of the English Language* (Unabridged). Springfield, Mass.: Merriam-Webster.

HINKLE, D. N. (1970). The game of personal constructs. In D. Bannister (ed.) *Perspectives in Personal Construct Theory.* London: Academic Press.

HOLTZMAN, S. H. (1989). *Intelligent Decision Systems.* Reading, Massachusetts: Addison-Wesley. In press.

HORVITZ, E. J., BREESE, J. S. & HENRION, M. (1988). Decision theory in expert systems and artificial intelligence. *Journal of Approximate Reasoning,* in press.

HOWARD, R. A. (1966). Decision analysis: Applied decision theory. In D. B. Hertz and J. Melese (eds.), *Proceedings of the Fourth International Conference on Operational Research.* Reprinted in Howard & Matheson, 1984.

HOWARD, R. A. & MATHESON. J. E. (1980). Influence Diagrams. Reprinted in Howard & Matheson, 1984.

HOWARD, R. A. & MATHESON, J. E. (eds.) (1984). *Readings on the Principles and Applications of Decision Analysis.* Menlo Park, California: Strategic Decisions Group.

JONES, J. C. (1981). *Design Methods.* New York: John Wiley & Sons.

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

KEENEY, R. L. (1986). *Value-driven Expert Systems For Decision Support.* Los Angeles: Systems Science Department, University of Southern California.

KEENEY, R. L. & RAIFFA, H. (1976). *Decisions With Multiple Objectives: Preferences and Value Tradeoffs.* New York: Wiley.

KELLY, G. A. (1955). *The Psychology of Personal Constructs.* 2 Volumes. New York: Norton.

KITTO, C. & BOOSE, J. H. (1987). Heuristics for expertise transfer: The automatic management of complex knowledge acquisition dialogs. *International Journal of Man-Machine Studies,* 26, 183-202.

LAING, R. D. (1965). Mystification, confusion, and conflict. In I. Boszormenyi-Nagy & J. L. Framo (Eds.), *Intensive Family Therapy: Theoretical and Practical Aspects.* New York: Harper and Row.

LAING, R. D. (1969). *Self and Others.* New York: Pantheon Books.

LASEGUE, C. & FALRET, J. (1877) La folie à deux, ou folie communiquée. *Annales Médico-Psychologiques,* 18. (English translation by R. Michaud, *American Journal of Psychiatry,* supplement to vol. 121, 4:2-18, 1964). Cited in Watzlawick & Weakland, 1977.

LIDZ, T., CORNELISON, A., TERRY, D. & FLECK, S. (1958). Intrafamilial environment of the schizophrenic patient: VI. The transmission of irrationality. *Archives of Neurology and Psychiatry,* 79: 305-315.

MARCUS, S. (1987). Taking backtracking with a grain of SALT. *International Journal of Man-Machine Studies*, 26, 383-398.

MCNAMEE, P. & CELONA, J. (1987). *Decision Analysis for the Professional with Supertree.* Redwood City, California: The Scientific Press, 1987.

MOORE, E. A. & AGOGINO, A. M. (1987). INFORM: An architecture for expert-directed knowledge acquisition. *International Journal of Man-Machine Studies*, 26, 213-230.

PEARL, J. (1985). How to do with probabilities what people say you can't. *Proceedings of the 2nd Annual Conference on Artificial Intelligence Applications,* Miami, December 11-13, 1985.

PEARL, J., GEIGER, P. & VERMA, T. (1988). The logic of influence diagrams. *Proceedings of the Conference on Influence Diagrams for Decision Analysis, Inference, and Prediction.* May 9-11, 1988, Berkeley California, University of California at Berkeley.

PEARL, J. & VERMA, T. (1987). The logic of representing dependencies by directed graphs. *Proceedings of the National Conference on Artificial Intelligence,* Seattle.

RAIFFA, H. (1968). *Decision Analysis: Introductory Lectures On Choices Under Uncertainty.* Reading, Massachusetts: Addison-Wesley.

RAPPAPORT, A. & GAINES, B. (1988). Integration of acquisition and performance systems. In *Proceedings of the AAAI Integration of Knowledge Acquisition and Performance Systems Workshop.* St. Paul, Minnesota, August 21, 1988.

REGE, A. & AGOGINO, A. M. (1988). Topological framework for representing and solving probabilistic inference problems in expert systems. *IEEE Transactions on Systems, Man, and Cybernetics,* 18(3), 402-414.

RUBINSTEIN, M. F. (1975). *Patterns of Problem Solving.* Englewood Cliffs, N.J.: Prentice-Hall.

RUSSO, P.J. (1988). Activity graphs. Internal technical memo.

SCHEFLEN, A.E. (1960). Regressive one-to-one relationships. *Psychiatric Quarterly,* 23:692-709.

SHAW, M. L. G. (Ed.) (1981). *Recent Advances in Personal Construct Technology.* London: Academic Press.

VON WINTERFELDT, D. & EDWARDS, W. (1986). *Decision Analysis and Behavioral Research.* Cambridge: Cambridge University Press.

WATZLAWICK, P. & WEAKLAND, J. H. (1977). *The Interactional View: Studies at the Mental Research Institute, Palo Alto, 1965-1974.* New York: W. W. Norton.

WATZLAWICK, P., WEAKLAND, J.H. & FISCH, R. (1974). *Change: Principles of Problem Formation and Problem Resolution.* New York: W. W. Norton.

WELLMAN, M. (1986). Reasoning about assumptions underlying mathematical models. In J. S. Kowalik (ed.), *Coupling Symbolic and Numeric Computing*. Amsterdam: Elsevier.

WISE, J. A. (1985). Decisions in design: Analyzing and aiding the art of synthesis. In G. Wright (ed.) *Behavioral Decision Making: Theory and Analysis.* New York: Plenum Press.

WYNNE, L.C., RYCKOFF, I.M., DAY, J. & HIRSCH, S.I. (1958) Pseudo-mutuality in the family relations of schizophrenics. *Psychiatry,* 21:205-220.

ZWICKY, F. (1969). *Discovery, Invention, Research through the Morphological Approach.* New York: Macmillan.