Design Knowledge Capture and Alternatives Generation Using Possibility Tables in *Canard*[1]

David B. Shema, Jeffrey M. Bradshaw, Stanley P. Covington, and John H. Boose
Advanced Technology Center, Boeing Computer Services
P.O. Box 24346, M/S 7L-64, Seattle, WA 98124-0346, (206) 865-3404, dshema@atc.boeing.com

ABSTRACT

During the evolution of a design concept, designers must integrate diverse sources and kinds of information about requirements, constraints, and tradeoffs. In doing so, they make certain assumptions and develop criteria against which alternatives are evaluated for suitability. Unfortunately, much of this process is implicit, making later review difficult if not impossible. When requirements change, impacts on the design are difficult to trace. This can lead to costly rework or serious errors. We are developing *Canard*, an automated tool which uses possibility tables, constraints, and knowledge bases to assist in the generation of design alternatives consistent with goals and constraints. The facility also attempts to capture and document assumptions and tradeoffs made during the design process. We present an example which illustrates the use of *Canard* for a simple configuration problem. A more complex example traces the activity of a Boeing expert building a possibility table for robot arm design. Finally, the application of *Canard* to a NASA corporate memory facility project is described.

1. INTRODUCTION

1.1. Problems of Design Knowledge Capture and Alternatives Generation

During the evolution of a design concept, designers must integrate diverse sources and kinds of information about requirements, constraints, and tradeoffs. In doing so, they make certain assumptions and develop criteria against which alternatives are evaluated for suitability. Unfortunately, much of this process is implicit, making later review difficult if not impossible. When requirements change, impacts on the design are difficult to trace. This can lead to costly rework or serious errors.

An automated tool provides at least documentation of the design history. The information used in defining the alternatives can be captured and stored in a retrievable manner. With such a capability, a more complete record of the design is available for later review and revision. The knowledge and experience of the designer is also available to others who may be facing a new problem that is similar to the old one.

[1] The work described in this paper was supported by NASA contract NAS2-12108.

In addition to preservation of the design history, we are concerned with helping designers better explore the space of alternatives. Cognitive scientists have long known that people typically retrieve only a small fraction of available alternatives when generating hypotheses (Wise, 1985). People 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, and Tversky, 1982).

Although it would be impossible in practice to guarantee that all relevant alternatives were considered, an automated tool could help designers develop a richer set of alternatives. By coupling the alternative generation facility to analysis tools, the system could also help designers identify alternatives that differ slightly from program requirements. The designer would get a better feel for the effects of the constraints on different alternatives and could better evaluate the consequences of assumptions and tradeoffs.

1.2. Approach to Building Tools for Automated Support

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

1.2.1. 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 with soft (optimal or “good enough”) constraints with respect to objectives (e.g., design, planning, configuration, scheduling). As Clancey (1984) observes, however, real-world problems do not always fall neatly into one of these two categories; generally the solution to complex problems involves the integration of several reasoning strategies.

A number of approaches have evolved to the solution of synthesis problems. More traditional approaches, from fields such as operations research, include optimization techniques (e.g., linear, nonlinear, and dynamic programming) and 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 (e.g., Marcus, 1987).

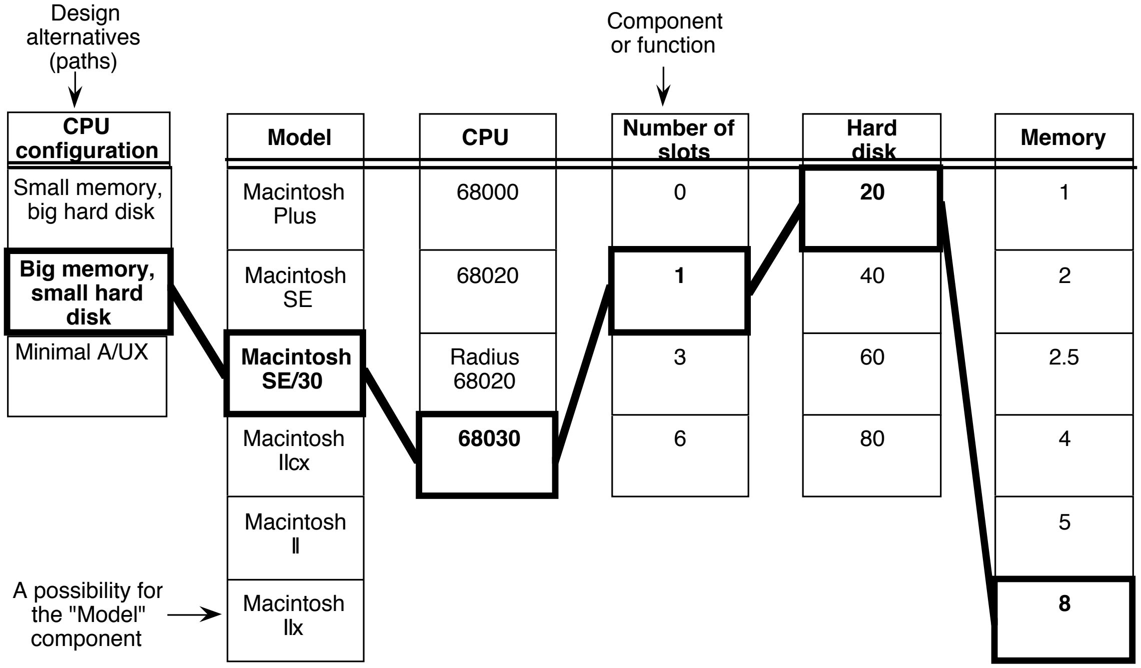
Traditional and knowledge-based approaches to solving synthesis problems are similar in many respects. Both usually begin with a statement of desired outcome states, in the form of constraints, and attempt to synthesize feasible or acceptable alternatives through exploration of the bounds and the interactions of the constraints. Since there will likely be several acceptable alternatives, the synthesis phase will be followed by analysis to determine which alternative best meets 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. We describe our view of the design process more completely in Bradshaw, Boose, Covington, and Russo (1989).

A tool known as a *possibility table* provides an integrating interface for the synthesis and analysis of design alternatives. Within this facility, components, constraints, and objectives can be synthesized into design alternatives with automated assistance. These alternatives are analyzed, using additional tools available within the interface, to make the consequences of a particular choice explicit and available for review at any time. We describe these facilities below.

1.2.2. Possibility Tables

The possibility table representation is not a new idea. Manually developed strategy tables have been used by decision analysts as one way of generating and representing complex alternatives (McNamee and Celona, 1987). Related approaches, such as Zwicky's morphological charts, (Zwicky, 1969) have been manually employed by designers for many years.

Possibility tables are used to structure information rabout complex alternatives, outcomes, or plans[1] (Figure 1). When used for a design or configuration problem, columns in the possibility table typically represent essential components or functions that compose the artifact being designed. Within each column, the various possibilities listed identify alternatives for that component. To construct a design alternative, a designer selects a possibility from each column, defining a unique path through the table. The names of these alternatives appear in the leftmost column.


Design alternatives (paths)
Component or function
A possibility for the "Model" component

CPU configuration	Model	CPU	Number of slots	Hard disk	Memory
Small memory, big hard disk	Macintosh Plus	68000	0	**20**	1
Big memory, small hard disk	Macintosh SE	68020	**1**	40	2
Minimal A/UX	**Macintosh SE/30**	Radius 68020	3	60	2.5
	Macintosh IIcx	**68030**	6	80	4
	Macintosh II				5
	Macintosh IIx				**8**

Figure 1. A possibility table for configuration of a Macintosh computer system.

[1] *Possibility table* is the generic term referring to the graphical representation. 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.

1.2.3. *Canard:* A Tool That Integrates Possibility Tables With Knowledge-based Tools in Aquinas and Axotl

Canard automates the possibility table representation and extends its logic and structure to allow knowledge-based inference and the representation of more complex problems than could be handled by manual approaches such as hierarchical tables or explicit representation of constraints. *Canard* receives assistance in alternatives generation and design knowledge capture through its integration with two additional tools: Aquinas, an automated knowledge acquisition workbench, and Axotl, a knowledge-based decision analysis and process management workbench (Figure 2).

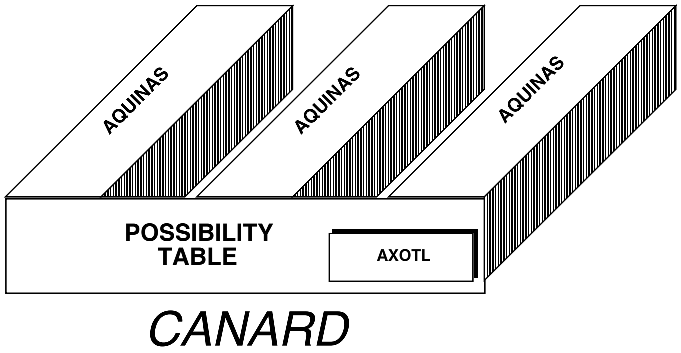


Figure 2. *Canard* is a design tool that integrates possibility tables with knowledge-based tools in Aquinas and Axotl.

Aquinas, an expanded version of the Expertise Transfer System (ETS) (Boose, 1984, 1985, 1986), is a workbench that supports several knowledge acquisition activities. The system interviews experts and helps them analyze, test, and refine a knowledge base. It uses methods from personal construct theory, an approach that grew out of George Kelly's research and experience as a clinical psychologist (Kelly, 1955). Activities supported by Aquinas include eliciting distinctions, decomposing problems, incrementally testing knowledge bases, integrating data types, automatically expanding and refining the knowledge base, using multiple sources of knowledge, and using constraints in the inference process. In the context of *Canard,* the tools in Aquinas are used to elicit and structure information about design component possibilities and gather the constraints and objectives that guide a designer in the selection of these possibilities. Analysis tools in Aquinas help designers determine the adequacy of the constraints and objectives and focus their attention on descriptions needing further refinement. Aquinas is discussed in greater detail in Boose and Bradshaw (1987) and Boose, Shema and Bradshaw (1988).

Axotl combines a decision analysis workbench with knowledge-based tools. The decision analysis workbench contains a graphical influence diagram editor (Howard and Matheson, 1980), which is used for creating and refining models of alternatives, preferences, and uncertainties relevant to a specific decision. The knowledge base tools in Axotl can be configured with application-independent knowledge (i.e., knowledge of decision analysis tools and methodology) and application-specific knowledge (i.e., knowledge about a particular domain) to provide guidance and help during a consultation. Within the *Canard* interface*,* the knowledge base tools in Axotl can be used to graphically represent and reason about constraints on a design and to represent knowledge about the design process itself. Additionally, the decision analysis workbench may be useful for decisions involving significant uncertainties, complex tradeoffs, or high stakes. *Axotl* is described in more detail in Bradshaw and Boose (1989) and Bradshaw, Covington, Russo, and Boose (1989).

Canard enhances possibility tables with constraint-handling tools which reduce the possible solution space and also capture important information about the design. *Canard* allows both hard and soft constraints. A designer adds hard constraints to possibilities to capture information about incompatibilities and interdependencies between component possibilities. During alternatives generation, these hard constraints prevent any incompatible components from being selected.

Soft constraints and utility scores may be added to attributes which characterize generated paths. The desired values of these attributes are the design goals. Conflicts between two or more design goals often necessitate making tradeoffs between the goals. For example, a goal of low cost is often in conflict with a goal of high reliability. Using *Canard,* the designer can specify the acceptable ranges of an attribute and map attribute values to the utility of these values. The system could then guide the designer toward possibilities that optimized the tradeoffs between the soft constraints.

Possibilities may be selected manually by the designer in an exploratory mode. However, for large problems it would be impractical to force designers to assign each possibility directly. In these cases, an iterative search proposes new alternatives based on permutations of the constraint space to assist in generating alternatives. Through a similar procedure, the system can hypothesize new constraints based on examples of previously defined alternatives. By keeping track of what has been tried before, the tool assists the designer in covering the possible solution space.

Possibility tables in *Canard* have a graphical interface which facilitates creation and modification of design information. The designer creates new components and possibilities and defines alternative paths using the mouse. The interface also assists the designer in managing the complexity of large design problems by presenting the design in an easily comprehensible view. Such an interface integrates well with the environments of *Aquinas* and *Axotl.*

The graphical possibility table facility in *Canard* is being implemented in ParcPlace Smalltalk-80 within the DDUCKS environment (Bradshaw, Covington, Russo, and Boose, 1989). DDUCKS supports coordination of conventional and knowledge-based applications running concurrently under MultiFinder on the Macintosh or across a network. This facility allows the possibility table tools in *Canard* to have access to all the functionality of *Aquinas* and *Axotl.*

2. EXAMPLE APPLICATIONS OF *CANARD*

To illustrate the use of *Canard,* we will describe a hardware configuration decision made recently as part of planning for a new project. Following this example, we present a more complex possibility table developed by a Boeing robotics expert. Finally, we will discuss how possibility tables will be used in more complex problems as part of a NASA corporate memory facility (CMF) study.

2.1. Personal Computer Configuration

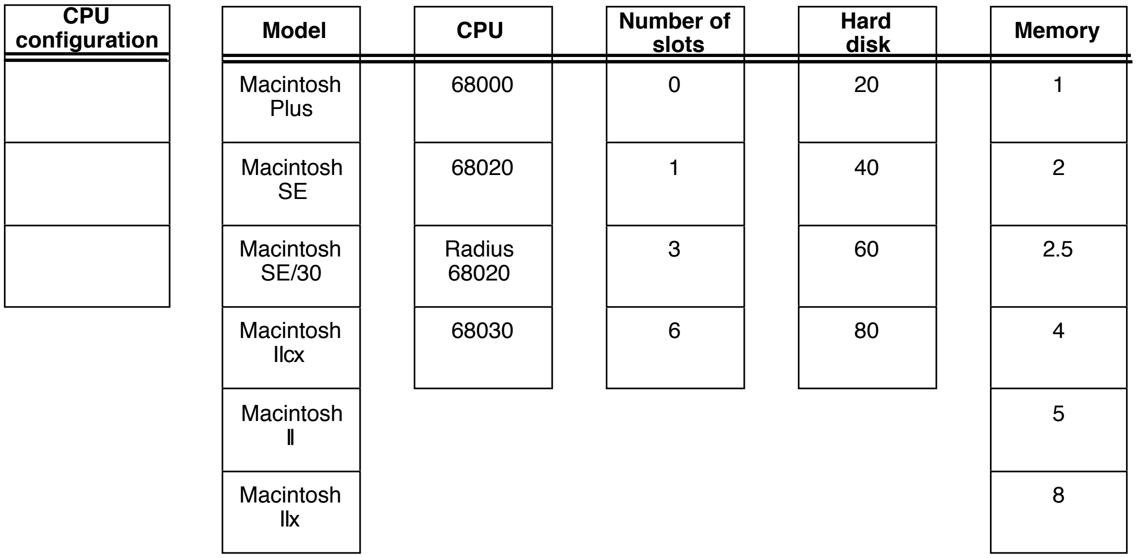
We recently began a new project which required the purchase of two or three Macintosh computer systems. Specific hardware requirements had to be met while remaining within a $15,000 budget. In addition, from the long-term perspective, team members were interested in configuring a system that would be expandable and flexible for future projects.

We will present the example in terms of seven different kinds of activities supported by *Canard*. The activities are highly interrelated, and it would be unreasonable to suppose that they would be followed in any sort of strict order. At any time, the designer might move from one activity to another according to the type of knowledge being entered into *Canard*. The designer might add or modify components, possibilities, and constraints at any time. The activities, described below, are:

1. Specifying components and possibilities.
2. Specifying inter-component compatibility constraints.
3. Specifying intra-component attributes, preferences, and constraints.
4. Specifying inter-component attributes, preferences, and constraints.
5. Proposing alternatives by specifying possibility paths.
6. Automating path generation.
7. Automating constraint generation.

2.1.1. Specifying Components and Possibilities

The designer first needs to build the basic structure of the possibility table. The initial components of the computer system are identified and become the column headers (Figure 3). Possibilities are listed under each component. The leftmost column is reserved for the names of the alternatives that will be generated.



CPU configuration	Model	CPU	Number of slots	Hard disk	Memory
	Macintosh Plus	68000	0	20	1
	Macintosh SE	68020	1	40	2
	Macintosh SE/30	Radius 68020	3	60	2.5
	Macintosh IIcx	68030	6	80	4
	Macintosh II				5
	Macintosh IIx				8

Figure 3. Components and possibilities are specified in the possibility table.

2.1.2. Specifying Inter-component Compatibility Constraints

Hard constraints are added to the table to capture information about incompatibilities between combinations of component possibilities. Compatibilities are stored in a tree-like structure; using tools from Axotl, the designer prunes the tree to remove the incompatibilities. Such a representation permits specification of complex incompatibilities between three or more interacting possibilities, as well as simple incompatibilities between pairs of possibilities. Simple incompatibilities may also be specified directly in the possibility table using a point-and-click mechanism. The designer selects combinations of incompatible possibilities, and *Canard* automatically prunes the compatibility tree.

A partial compatibility tree, showing the CPU, memory, and hard disk configurations available for the Macintosh SE, is shown in Figure 4. The single hard disk branch is unlabeled and unexpanded, indicating that all hard disk possibilities are compatible with every other type of possibility shown. The constraints specified in the compatibility tree cannot be relaxed.

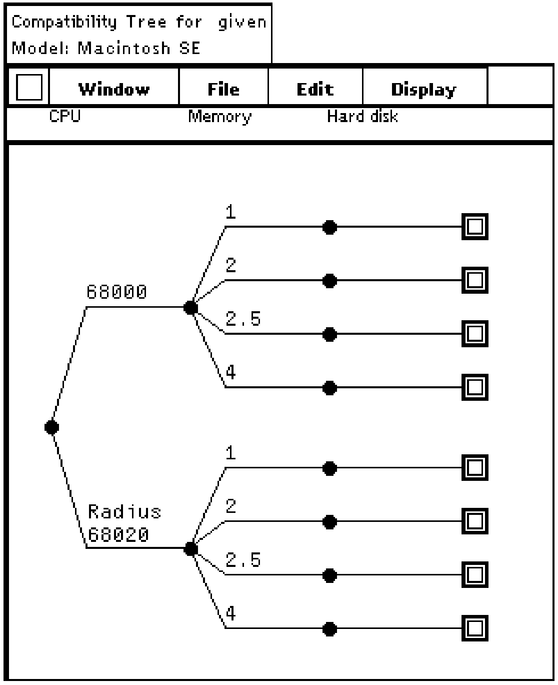


Figure 4. A partial compatibility tree, showing the CPU, memory, and hard disk configurations available for the Macintosh SE.

2.1.3. Specifying Intra-component Attributes, Preferences, and Constraints

The designer creates an Aquinas knowledge base for each component. Within Aquinas*,* a number of useful representation, interviewing, and analysis techniques have been developed to help designers determine distinguishing features of component possibilities. Many of these techniques are based on the research of George Kelly (1955), a clinical psychologist who emphasized the foundational role of distinctions (*personal constructs*) underlying the processes of perception and reasoning. For example, using a *triadic elicitation* interviewing technique, Aquinas asks designers to define similarities and differences by considering groups of component possibilities presented three at a time: “Think of an important attribute that two of Macintosh Plus, Macintosh SE, and Macintosh SE/30 share, but that the other does not. What is that characteristic?” After giving Cost

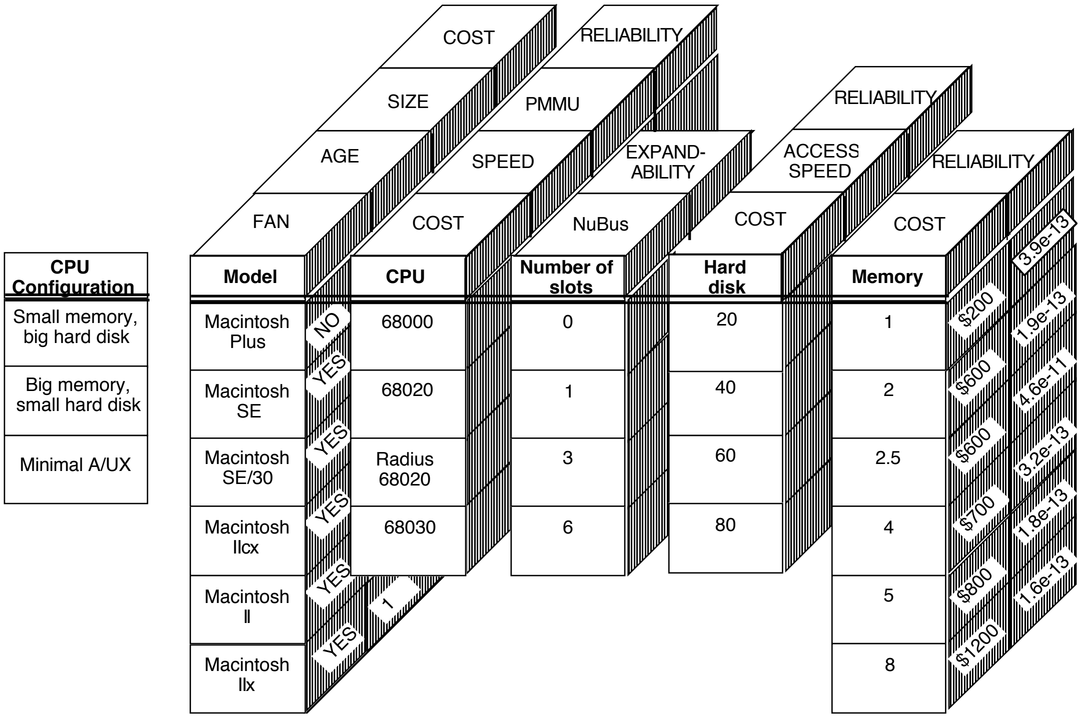
(More Money/Less Money) as that characteristic, the person might then be asked about Macintosh SE, Macintosh SE/30, and Macintosh IIcx and come up with a second characteristic Size (Small-Medium-Large), and so forth.

One way of representing this information is through a *repertory grid* (Figure 5), a matrix with elements (i.e., component possibilities) ranged along the bottom and constructs (i.e., dimensions of similarity and difference between elements), defined by extension, as a horizontal row of values (or probability distributions) within the matrix. The grid presentation allows the user to see patterns of similarity and difference that would otherwise be difficult to grasp. Analysis techniques in Aquinas (e.g., similarity analysis, cluster analysis) exploit these patterns to help users discriminate more carefully among similar concepts as part of model refinement. Implication analysis helps users discover important dependencies between constructs.

Macintosh Plus	Macintosh SE	Macintosh SE/30	Macintosh IIcx	Macintosh II	Macintosh IIx	
N	Y	Y	Y	Y	Y	*Fan (Yes-No)*
5	3	0	0	2	1	*Age (Older-Newer)*
S	S	S	M	L	L	*Size (Small-Medium-Large)*
1.1	1.9	2.5	2.9	2.8	4.0	*Cost (Less Money-More Money)*

Figure 5. Model possibilities are compared and contrasted along attributes defined using *Aquinas*.

The grid can be viewed as being "plugged in" to the back of a column of the possibility table (Figure 6) and accessible to the possibility table. The attributes and possibilities for a particular column "slice" form the rows and columns of the repertory grid, and the values for each possibility with respect to each attribute constitute the grid ratings.



CPU Configuration
Small memory, big hard disk
Big memory, small hard disk
Minimal A/UX

Model
Macintosh Plus
Macintosh SE
Macintosh SE/30
Macintosh IIcx
Macintosh II
Macintosh IIx

CPU
68000
68020
Radius 68020
68030

Number of slots
0
1
3
6

Hard disk
20
40
60
80

Memory
1
2
2.5
4
5
8

Figure 6. Aquinas knowledge bases "plug into" the back of the possibility table.

The knowledge base is used during the design process to guide the designer in selecting the best possibilities for each component. Hard constraints filter out incompatible possibilities. Tradeoffs are computed from grid information, given each attribute value for each component possibility and the preferred attribute values (soft constraints) for each component, resulting in overall utility scores for each possibility.

Constraints and preferences for each attribute can be entered using an interactive plotting facility. Designers use the mouse to assign relative utility to the range of values for an attribute (Figure 7). Utilities between 1 and 100 are used as soft constraints in the selection of the best possibility for each component. If a particular possibility has a rating that maps to a utility of 0, *Aquinas* treats the information as a hard constraint and eliminates the possibility from further consideration. For example, in Figure 7 the designer has asserted that any model with a base price of greater than $3,500 has a utility of zero. This eliminates the Macintosh IIx, with a cost of $4000, from further consideration (see Figure 5). In the possibility table, the Macintosh IIx entry will be inverted to signify that it will be excluded from further consideration (see Figure 9). Constraints can also be entered using a constraint language.

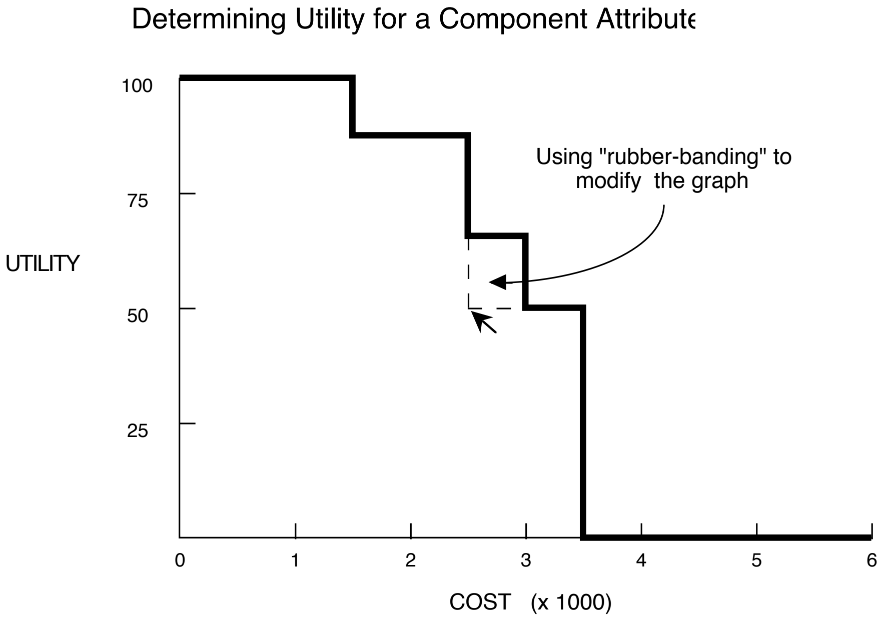


Figure 7. Graphical facility allows attribute values to be mapped to a scale of utility.

Using Aquinas, relative weights are assigned to the attributes and used to compute an overall utility score for each possibility. As information is changed in the grid, the utility score can be dynamically computed and displayed next to the possibilities in the possibility table. This score will guide designers in selecting the best possibility within a column, given previous selections, when they construct paths through the table.

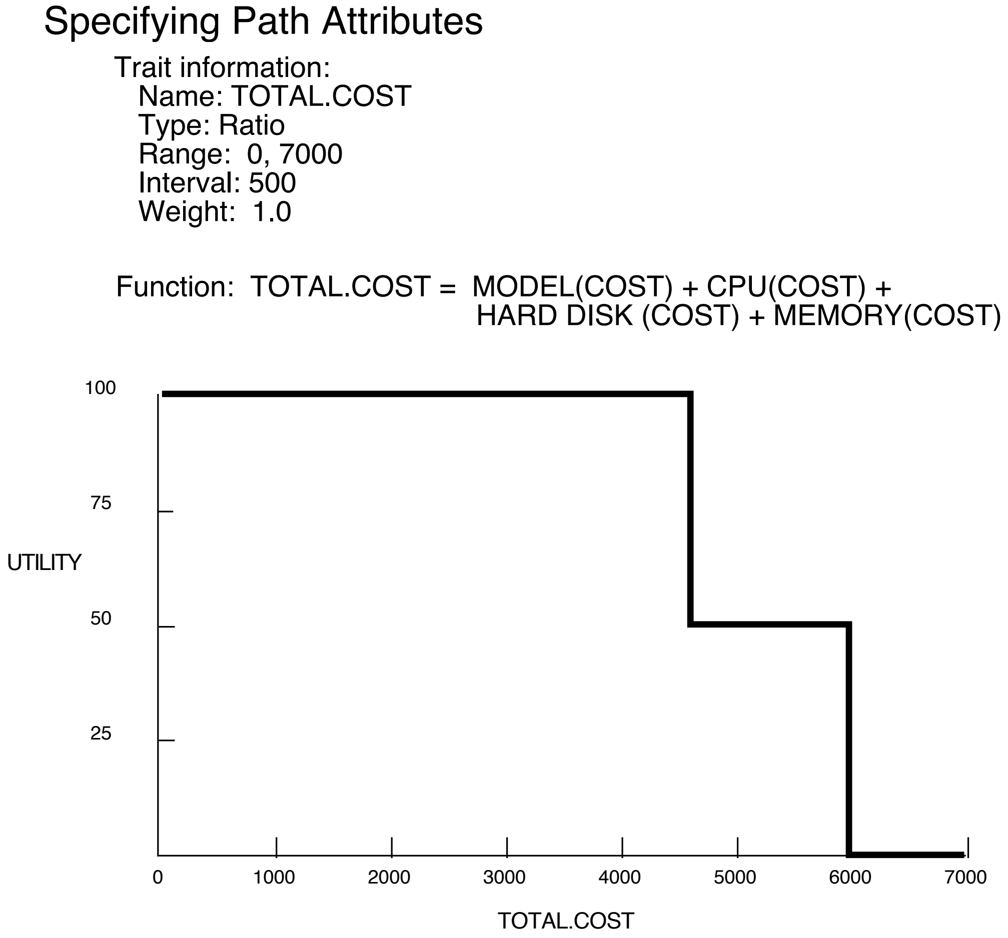
2.1.4. Specifying Inter-component Attributes, Preferences, and Constraints

Inter-component (or path) attributes, preferences, and constraints characterize the possibility paths that define each alternative. The attributes are important dimensions shared across some subset of the components. Preferences on these inter-component attributes specify the overall goals which should be optimized for the design. (A utility metric measures the degree to which an attribute value satisfies a preference – the higher the utility, the better the alternative.)

The designer first provides the name, scale type (nominal, ordinal, interval, or ratio), range, interval, and weight information for each attribute. Next, the designer defines a function that will calculate the inter-component attribute value. During the specification of a path, the function will dynamically update the total value of the path attribute as the possibilities are selected. The current value is displayed in an active gauge (see Figure 9).

To see how well the alternative satisfies overall design goals, designers map inter-component attribute values to a utility scale in the same manner as for intra-component attributes (Figure 8; compare to Figure 7).

In Figure 8, the designer has defined the path attribute TOTAL.COST. The range of TOTAL.COST is 0 to 7000 in intervals of 500. The weight of 1.0 specifies the relative contribution of TOTAL.COST to the overall path utility score in comparison with other path attributes. On the utility graph, the designer has indicated that costs between 0 and 4500 are equally preferable. Costs between 4500 and 6000 are acceptable, but not nearly as desirable. A cost greater than 6000 has zero utility: the designer wants to avoid such a cost, if possible.



Specifying Path Attributes

Trait information:
Name: TOTAL.COST
Type: Ratio
Range: 0, 7000
Interval: 500
Weight: 1.0

Function: TOTAL.COST = MODEL(COST) + CPU(COST) + HARD DISK (COST) + MEMORY(COST)

Figure 8. The designer provides information about the TOTAL.COST attribute.

2.1.5. Proposing Alternatives by Specifying Possibility Paths

An alternative may be generated by selecting a possibility from each component using the mouse. As each possibility is selected, incompatible possibilities in other components are inverted to signal their incompatibility with hard constraints on the possibilities already selected. Graying of possibilities signifies level of desirability with respect to soft constraints. Active gauges associated with the path attributes display the current values of their function and a utility score. If a value of a path attribute exceeds the limit specified in a hard constraint, the designer may need to retract one or more possibility selections.

In Figure 9, the designer is in the process specifying a path for the alternative "*Small memory, big hard disk* " through the possibility table. *Model*, *Slot*, and *Disk* components have already been selected and are outlined. The inverted possibilities indicate that the possibilities that were found to be incompatible when the *Model* possibility was selected. The *Model* possibility *Macintosh IIx* was eliminated prior to path specification because of the hard constraint on CPU cost. The table and active gauges associated with the path attributes TOTAL COST and AVE RELIABILITY display the values of the cost and reliability functions as well as their utility score.

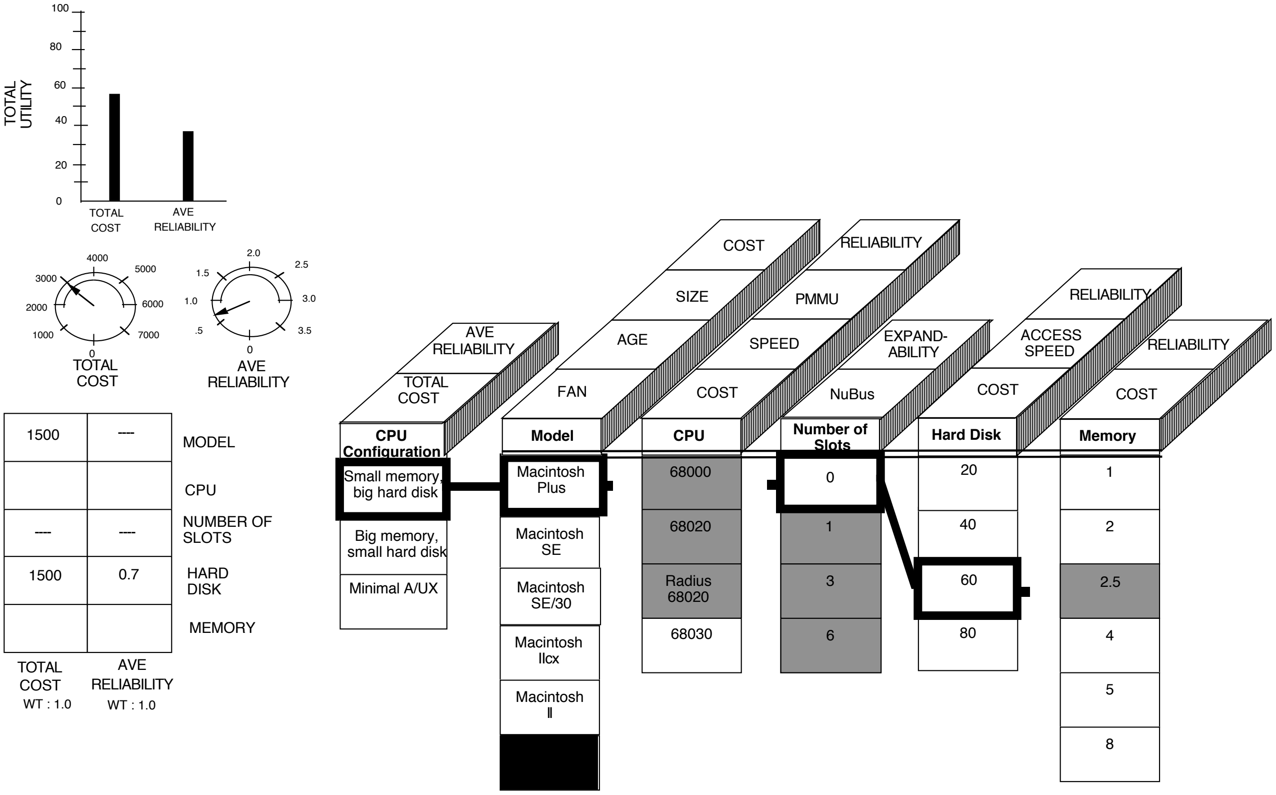


Figure 9. Specifying a possibility path to define an alternative.

Labeling the specified path is in and of itself an interesting part of the alternatives generation process. The designer may supply an initial label that describes the design concept being defined and then modify the description as the path is specified. As the designer experiments with different possibilities, naming the concepts embodied in the new paths helps the designer gain additional insights about the problem.

2.1.6. Automating Path Generation

Knowledge of the attributes, constraints, and preferences of components and possibilities permits *Canard* to automatically propose optimal paths through the table. It examines existing constraints and preferences and generates new paths that avoid component incompatibilities and maximize the combined utility of inter-component and intra-component attributes. Through this process, designers may be led to explore alternatives that might have otherwise gone unconsidered. The automatically generated path is presented to the designer, who may accept it or reject it. If it is accepted, the new alternative can be labeled and considered in more detail. If a proposed path is rejected, the designer is asked to state a rationale for rejection in the form of new or modified constraints.

2.1.7. Automating Constraint Generation

When a number of good alternative paths have been specified, *Canard* can analyze existing paths and query the designer about combinations of component possibilities that have not been explored. If the designer can explain why the combination has not been explored previously, a new constraint has been discovered. Otherwise, a new alternative can be proposed.

2.2. Robot Design

An engineer at Boeing with a background in robotics was asked to create a possibility table that could assist with the design of a robot arm for use on a vehicle such as Space Station Freedom or the Space Shuttle. Because *Canard* was not fully implemented at the time, the possibility table was constructed manually. We include the example to illustrate what *Canard* shall be capable of. The engineer first named three of the components used to construct an arm (*Internal Power, End Effector, Arm Base Type*) and supplied possibilities for each component. Asked to differentiate between the possibilities for *Arm Base Type,* he identified two attributes: Mobility and Rigidity. (Figure 10.)

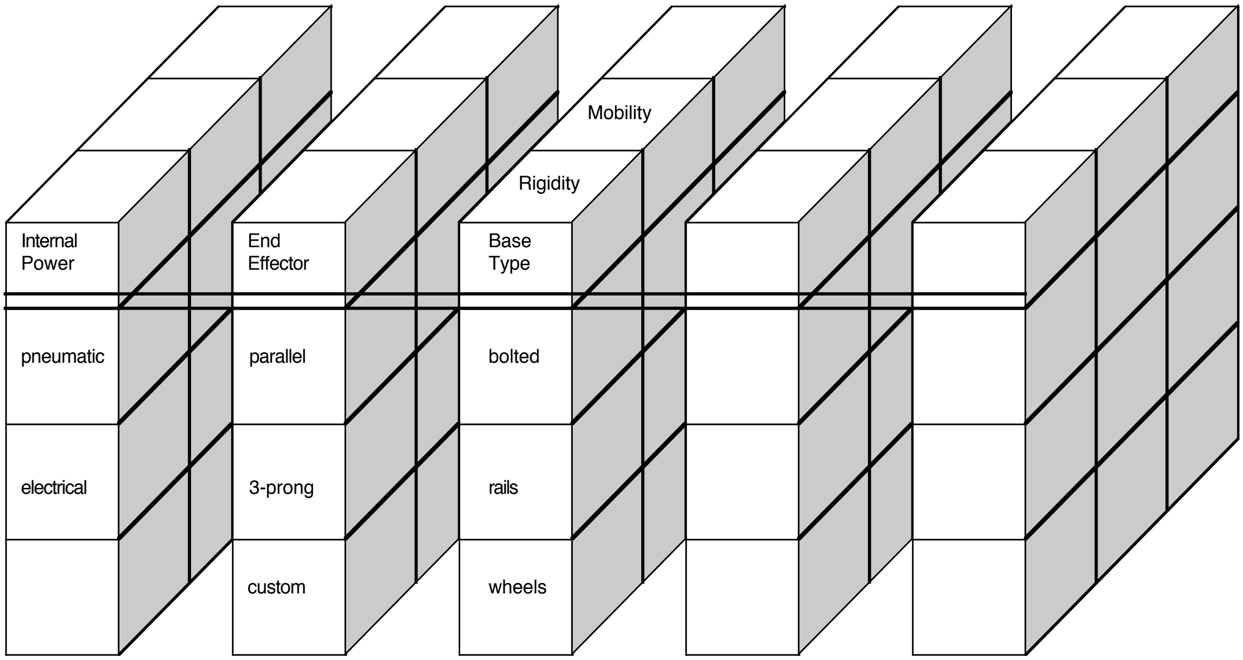


Figure 10. Robot arm possibility table under construction

The engineer supplied additional components for the construction of a robot arm and gave a few possibilities and attributes for each component. He then tried constructing a path for a simple robotic task (stacking blocks) and realized that his *Internal Power* component was no longer needed, since he had added *Motor-Power Arm* and *Motor-Power End-Effector* components. The unneeded component was deleted and the new path completed (Figure 11).

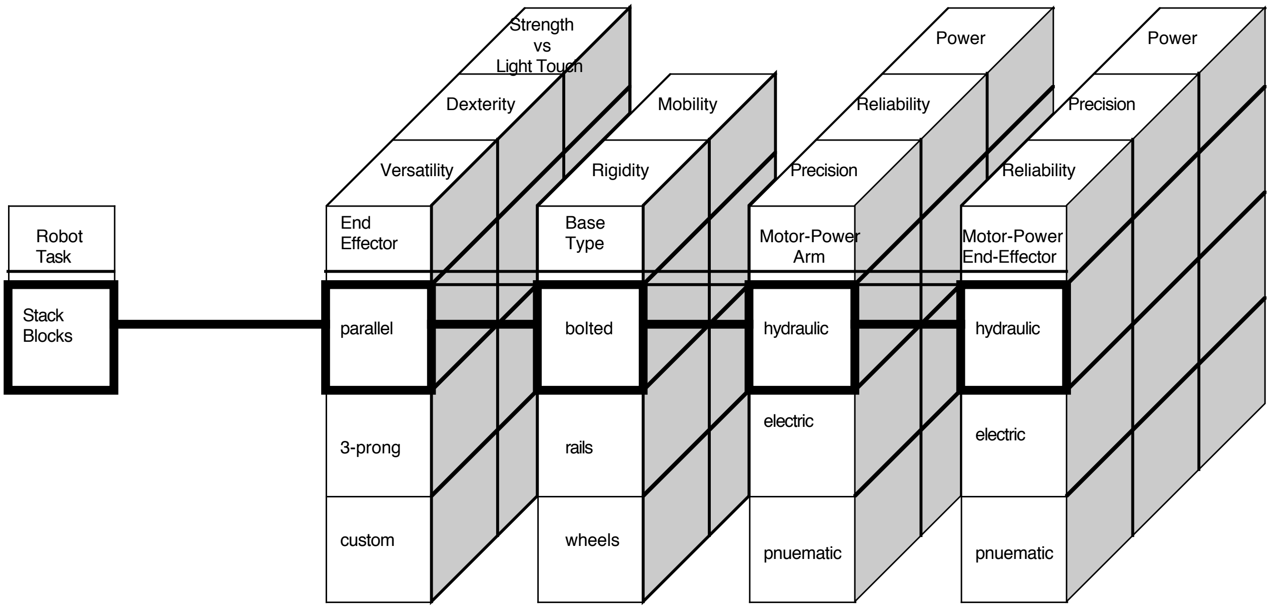
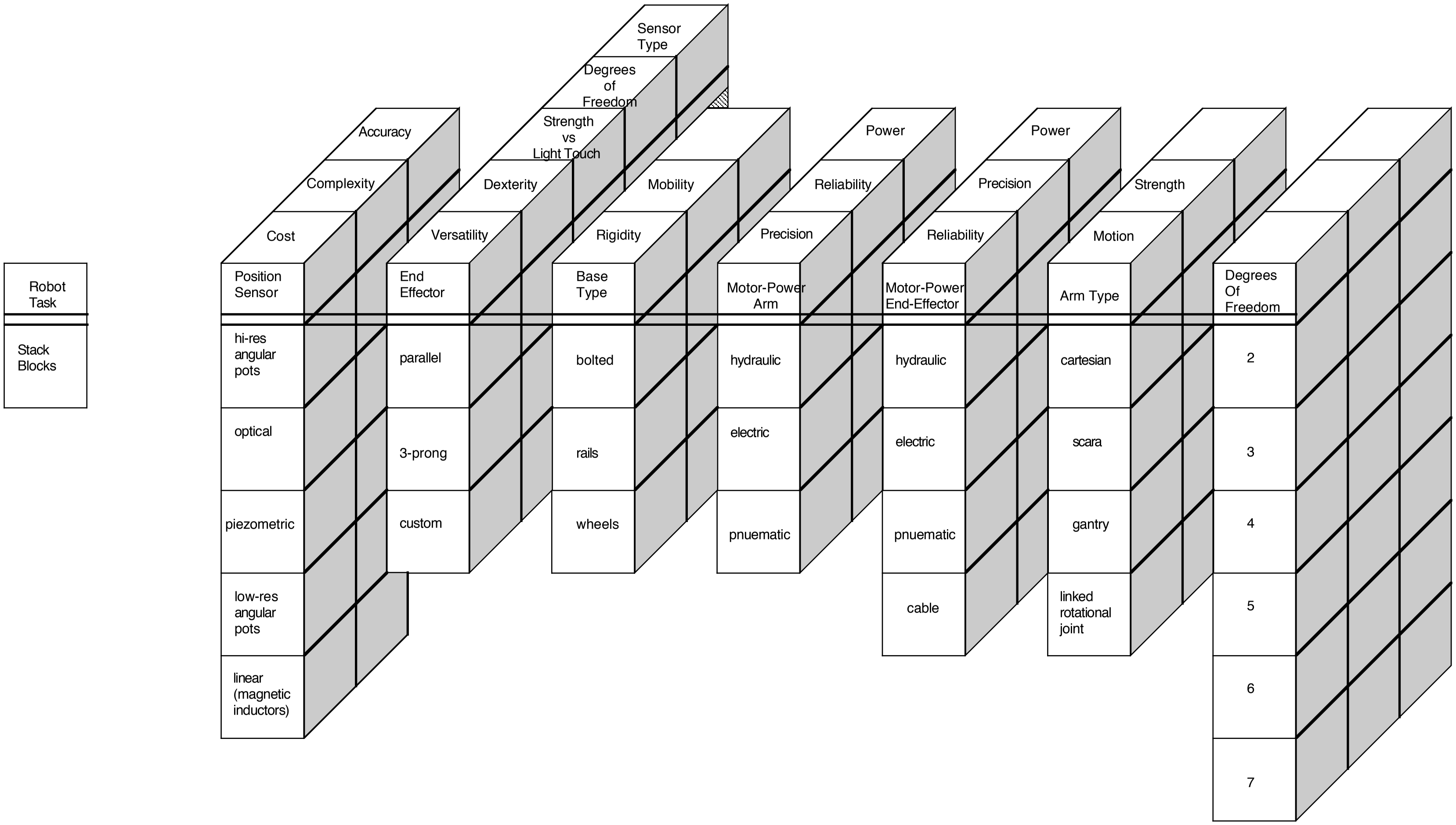


Figure 11. Selecting a Path for a Simple Task

As he was selecting a possibility from each component, he realized that some important components were missing. *Position Sensor*, *Arm Type*, and *Degrees of Freedom* were added, along with a set of possibilities and attributes for each component. The *Degrees of Freedom* component caused some discussion. Unlike the other components, this was not an actual part. However, the expert believed that it was an integral part of the design and should be included (Figure 12).

When asked to select a path for the design of a robot arm for use as a space station arm, the expert had little difficulty in choosing possibilities, with the exception of the *End Effector* component. None of the possibilities for that component were suitable, so he supplied a new possibility (Figure 13).

In the last exercise, the expert was given a partial path of possibilities from three components and asked if he could come up with a robot task naming such a path. Constructing a path from possibilities that are rarely used is a technique that can be used to ensure tht the design space is more fully covered (Figure 14).



Robot Task
Stack Blocks

Position Sensor	End Effector	Base Type	Motor-Power Arm	Motor-Power End-Effector	Arm Type	Degrees Of Freedom
hi-res angular pots	parallel	bolted	hydraulic	hydraulic	cartesian	2
optical	3-prong	rails	electric	electric	scara	3
piezometric	custom	wheels	pnuematic	pnuematic	gantry	4
low-res angular pots				cable	linked rotational joint	5
linear (magnetic inductors)						6
						7

Figure 12. The possibility table after adding the components *Position Sensor*, *Arm Type* and *Degrees of Freedom*.

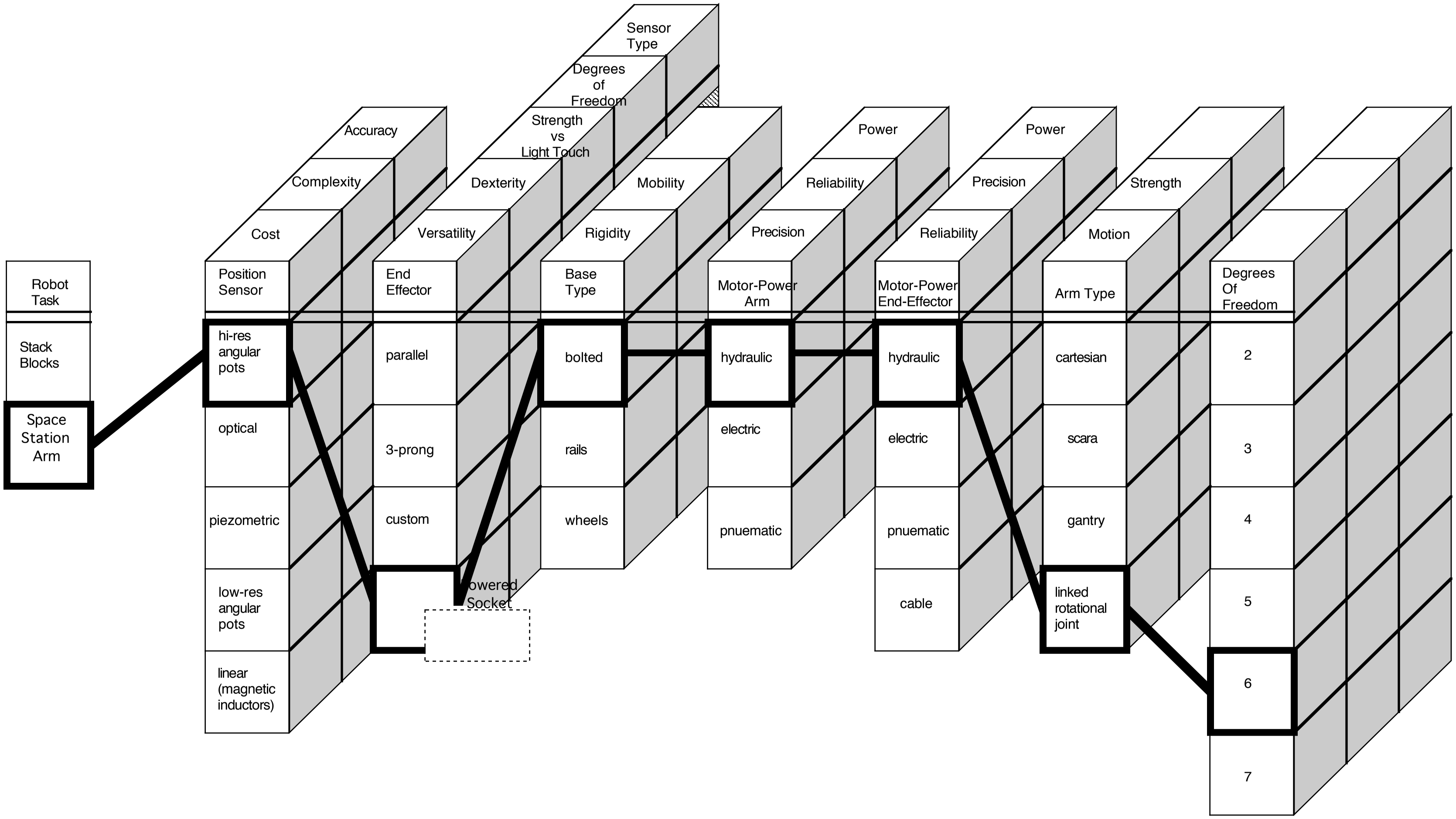


Figure 13. The engineer constructed this possibility path for *Space Station Arm.*

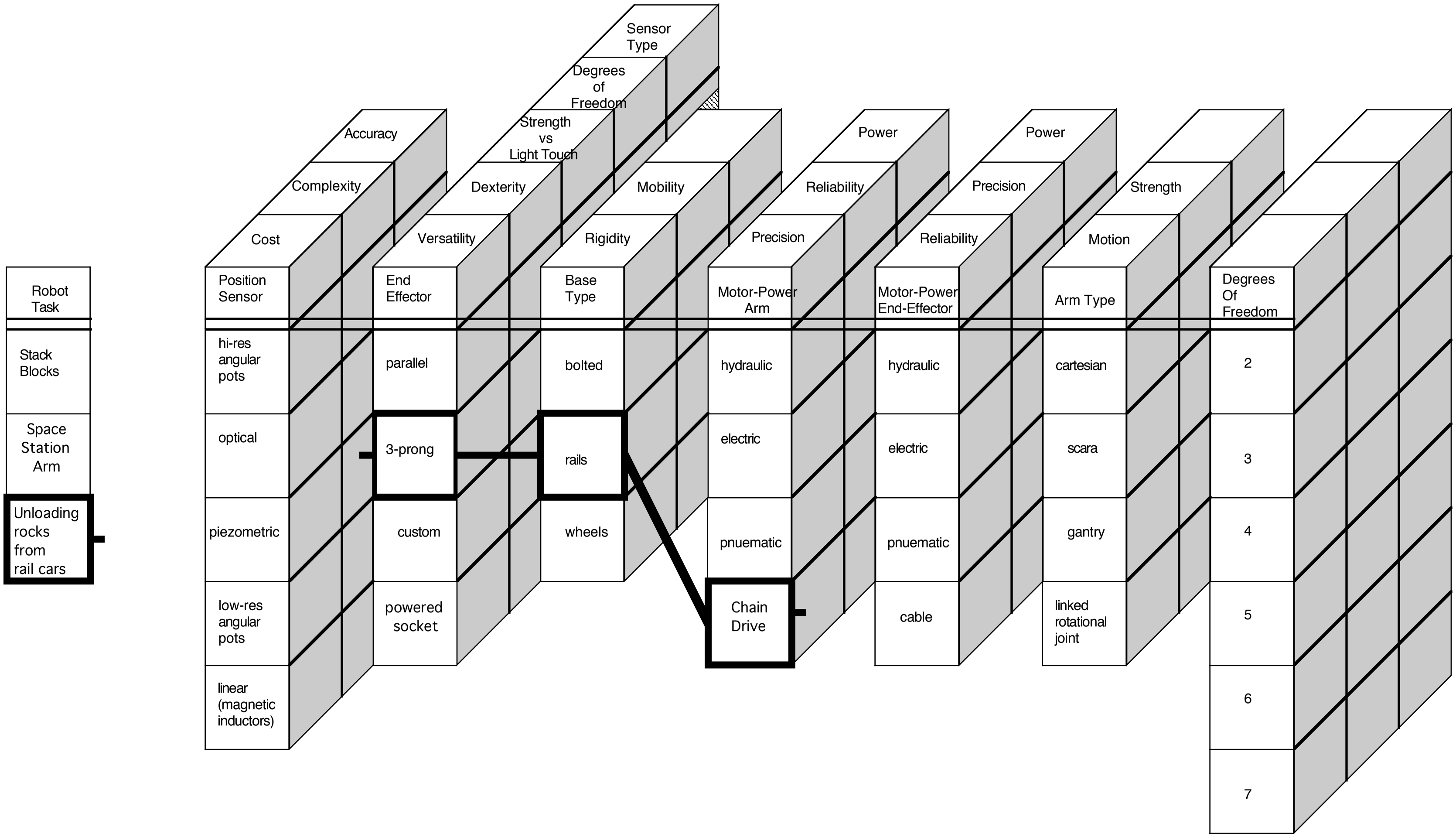


Figure 14. The engineer generates a new alternative consistent with a partial path proposed by the system.

2.3. APPLICATION OF *CANARD* TO NASA CORPORATE MEMORY FACILITY PROJECT

Under a NASA contract, the Boeing Advanced Technology Center is conducting research leading to a corporate memory facility (Boeing Computer Services 1989a,b,c). The goal of this project is to study and demonstrate techniques that could be employed to capture and use decision history and rationale throughout the life cycle of a major NASA program, such as Space Station Freedom. Within the Space Station Freedom program, we are examining aspects of the Power subsystem and the Environmental Control and Life Support subsystem. We are studying two areas to see if our ideas will apply across a breadth of applications, in different design stages, solving different types of problems.

We are focusing our Phase 1 research on design knowledge capture. NASA believes a Design Knowledge Capture tool should assist the engineer in providing a comprehensive definition of design knowledge including the rationale for the design chosen, specification of the design's performance, and functional breakdowns of the design showing dependencies and interrelationships (NASA, 1988). As an important part of this project, *Canard* is intended to demonstrate the feasibility of using an automated tool to generate and capture some of the information being considered at various times during the design process.

3. STATUS AND FUTURE WORK

The current version of *Canard* contains the basic graphical components for constructing and displaying possibility tables and paths, a rudimentary link to *Aquinas* and *Axotl,* and a limited set of constraint tools. Future work will include enhancing these basic capabilities to the extent described in this paper and performing a thorough evaluation of the tools in the context of NASA design problems. We also contemplate the eventual implementation of other features, including:

Hierarchies of possibility tables. The alternatives and path attributes constructed at one level become components in a possibility table at a higher level of design.

Voice and graphic explanations. Additional information about the possibilities, alternatives, traits, and design goals can be added by the designer in text, graphics, or voice format.

Design change history. As changes are made to the paths during the process of making tradeoffs, *Canard* will record the changes and prompt the designer for reasons for the changes. Voice or text formats will be available.

Links to external databases or knowledge bases, and model-based reasoning tools. External databases containing component information and knowledge bases previously developed can be plugged into the possibility table alongside the *Aquinas* knowledge bases.

Links to Axotl influence diagram facility. Influence diagrams could be plugged into the back of possibility tables in the same manner as repertory grids, to assist in problems with significant risks or uncertainties or complex tradeoffs.

Multiple expert tools. Aquinas tools that assist multiple experts in reaching consensus could be integrated with the *Canard* facility (Boose and Bradshaw, 1987).

ACKNOWLEDGEMENTS

We express our appreciation to Miroslav Benda, Kathleen Bradshaw, Beverley Clark, Karolyn Covington, Mike Freeman, Peter Friedland, Randy Johnson, Cathy Kitto, Sandra Marcus, Guy Olney, Kish Sharma, Suzanne Shema, Greg Swietek, Lisle Tinglof-Boose, and Bruce Wilson for their contributions and support. *Canard, Aquinas*, *Axotl,* and *DDUCKS* are under development at the Advanced Technology Center, Boeing Computer Services in Seattle, Washington.

REFERENCES

BOEING COMPUTER SERVICES (1989a) Information to be Retained in a Corporate Memory (CMF TR1), NASA contract NAS2-12108.

BOEING COMPUTER SERVICES (1989b) Knowledge Representation for a Corporate Memory Facility (CMF TR2), NASA contract NAS2-12108.

BOEING COMPUTER SERVICES (1989c) Knowledge Representation and Trade Studies (CMF DEMO1), NASA contract NAS2-12108.

BOOSE, J. H. (1984). Personal construct theory and the transfer of human expertise. *Proceedings of the National Conference on Artificial Intelligence,* Austin, Texas.

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). A survey of knowledge acquisition techniques and tools. *Proceedings of the Third AAAI Knowledge Acquisition for Knowledge-Based Systems Workshop,* Banff, Canada, November 1988. To appear in *International Journal of Man-Machine Studies.*

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. & BOOSE, J.H. (1989). Decision analysis techniques for knowledge acquisition: Combining information and preference models using *Aquinas*. *International Journal of Man-Machine Studies,* in press.

BRADSHAW, J.M., BOOSE, J.H., COVINGTON, S.P. & RUSSO, P.J. (1989). How to do with grids what people say you can't: The application of decision analysis methods in *Axotl* and personal construct methods in *Aquinas* to design problems. *Knowledge Acquisition: An International Journal,* in press.

BRADSHAW, J.M., COVINGTON, S.P., RUSSO, P.J. & BOOSE, J.H. (1989). Knowledge acquisition techniques for intelligent decision systems: Integrating *Axotl* and *Aquinas* in *DDUCKS*. *Proceedings of the AAAI Uncertainty Workshop*, August 18-20, Windsor, Ontario, Canada.

CLANCEY, W. J. (1984). Classification problem solving. *Proceedings of the National Conference on Artificial Intelligence,* Austin, Texas.

HOWARD, R. A. & MATHESON. J. E. (1980). Influence Diagrams. Reprinted in Howard & Matheson (eds.) (1984). *Readings on the Principles and Applications of Decision Analysis*. Menlo Park, California: Strategic Decisions Group.

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

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

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.

NASA (1988). Process Requirements Document for Design Knowledge Capture, Space Station Program, March 28, NASA Parkridge Space Center, Reston, VA.

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

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.

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