TECHNICAL REPORT

Sibyl: A Tool for Managing Group Decision Rationale

August, 1990

Jintae Lee

ccstr#113 sswp#3189

CENTER FOR
COORDINATION
SCIENCE

Massachusetts Institute of Technology
Sloan School of Management
Cambridge, Massachusetts

Sibyl: A Tool for Managing Group Decision Rationale

August, 1990

Jintae Lee

ccstr#113 sswp#3189

SIBYL: A Tool for Managing Group Decision Rationale

Jintae Lee
(jin@ai.mit.edu)

Center for Coordination Science,
and Artificial Intelligence Laboratory, MIT
E40-140, 1 Amherst St
Cambridge, MA 02139

ABSTRACT

We describe SIBYL, a system that supports group decision making by representing and managing the qualitative aspects of decision making processes: such as the alternatives, the goals to be satisfied, and the arguments evaluating the alternatives with respect to these goals. We use an example session with SIBYL to illustrate the language, called DRL, that SIBYL uses for representing these qualitative aspects, and the set of services that SIBYL provides using this language. We also compare SIBYL to other systems with similar objectives and discuss the additional benefits that SIBYL provides. In particular, we compare SIBYL to gIBIS, a well-known "tool for exploratory policy discussion", and claim that SIBYL is mainly a knowledge-based system which uses a semi-formal representation, whereas gIBIS is mainly a hypertext system with semantic types. We conclude with a design heuristic, drawn from our experience with SIBYL, for systems whose goal includes eliciting knowledge from people.

Explicit representation of a decision rationale, i.e. the deliberations leading to a decision, can provide many potential benefits, especially in the context of group decision making. The knowledge that people bring to a decision becomes available for others to augment or respond. The representation of a decision making process serves as a documentary record of how the decision develops, which in turn can serve as a basis for learning and justification. [Yakemovic and Conklin 90] provides a good documentation of these benefits and their ramifications. In addition, if the representation is well-structured, the system can provide services, such as managing dependencies among what is represented, that help people make better decisions. To achieve these benefits, the language for representing decision rationales should allow people to express naturally what they need to express, and people should get rewarded for using the language in their decision making. Most existing languages for representing decisions, such as decision trees [Raiffa 68] and influence diagrams [Howard and Matheson 81], are not designed to represent the deliberation aspect of decision making, but only the results of such deliberations. The few whose goal is to represent the decision rationale, such as gIBIS [Conklin and Begeman 88], are not either expressive enough and/or do not provide enough services to reward the user , as we discuss below. In this paper, we describe a system, called SIBYL* , which is designed to meet these requirements by providing a language structured for the task of decision making and the set of services that rewards the users of this language.

We proceed as follows. First, we elaborate the motivations underlying SIBYL. We then describe SIBYL; we briefly describe the language, called DRL (Decision Representation Language), that SIBYL uses for representing decision processes, and illustrate its use in an example session with SIBYL. After describing SIBYL, we compare SIBYL to other systems with similar objectives, especially to gIBIS, a hypertext system whose goal is to record design rationales. We discuss how SIBYL is similar to gIBIS, how SIBYL extends gIBIS, and what we gain as a result. We conclude with a design heuristic, drawn from our experience with SIBYL, for systems which have to elicit knowledge from people to provide its services.

MOTIVATIONS

There are two major motivations underlying SIBYL: knowledge sharing and qualitative decision support. These motivations also help us delimit the domain of application for SIBYL. SIBYL is useful in the situations, such as in design decisions, where the following motivations exist.

Knowledge Sharing: Decision making usually involves gathering and relating pieces of knowledge relevant to evaluating alternatives along some criteria. Such knowledge, once explicitly represented, can be shared by others in several ways. In a group decision making, explicitly represented knowledge allows people to augment the knowledge by bringing in additional knowledge, supporting claims, denying claims, or qualifications. We describe below how SIBYL allows this mode of knowledge sharing. Another mode of knowledge sharing takes place across groups. The knowledge represented as a part of a decision process is often useful to others making similar decisions. Past decisions can tell us not only the factual information that we need but also the ways that a decision can be structured, the ways that different goals can be achieved, or the attributes against which alternatives should be evaluated. Furthermore, past decisions provide the additional knowledge of whether the approach they took were successful or not. Yet another way of sharing knowledge is within a group across time. The records of how decisions were made serve as documents, which in turn serve as a basis for justification and learning. SIBYL also helps sharing knowledge through these modes, which we discuss only briefly when describing its services.

Qualitative Decision Support: Once we have a language for representing the qualitative structure of decision making processes, the system can provide many services that support decision making. For example, the system can manage the dependencies among objects. It can propagate the effect or uncertainty of additional knowledge. The system can keep track of multiple viewpoints. And it can retrieve from past decisions pieces of knowledge useful for the current decision. We describe these services in Section 3.

SIBYL

SIBYL consists of three parts: DRL (Decision Representation Language), a set of services that provides qualitative decision support by using what is represented in DRL, and the

* A Sibyl was one of a number of prophetesses in Greek mythology who gave wise counsel to people for their decision making.

user interface that makes it easier for people to use DRL. We discuss the details of DRL and the services elsewhere [Lee 90]. In this paper, we illustrate them in the context of an example session with SIBYL, thus describing SIBYL from the user's perspective. First, however, we describe only the bare essentials of DRL necessary to make our example session comprehensible. We then present an example session with SIBYL that illustrates how people contribute their knowledge to a decision making process. In the last subsection, we discuss the services that SIBYL can provide using the knowledge represented in DRL.

DRL (Decision Representation Language)

The objects and the relations that form the vocabulary of DRL are described below and shown graphically in Figure 1. Figure 2 shows a decision graph, i.e. a network representation of the DRL knowledge base, that we will use in our example session with SIBYL.

Alternatives represent the options from which to choose. Goals represent the properties that an ideal option should have. A Decision Problem represents the problem of choosing the Alternative that best satisfies the Goals. Each Alternative is related to a Goal via an Achieves relation, denoted as Achieves(Alternative, Goal). A relation in DRL is a subclass of Claim; in particular, the relation Achieves(A,G) represents the claim that the alternative A achieves the goal G. The overall evaluation of an alternative is represented by the plausibility of the relation, i.e. the claim, Is-the-Best-Alternative-For(Alternative, Decision Problem). The plausibility of this relation, in turn, is a function of the plausibility of the Achieves relations between the alternative and all the goals as well as of the importance of these goals. An Alternative is evaluated by arguing about the plausibility of the Achieves claims linking the alternative to each of the Goals, and about the importance of the Goals. More generally, one argues in DRL by producing a Claim, which can Support, Deny, or Presuppose other Claims. These relations -- Supports, Denies, Presupposes -- are, as mentioned above, claims; as such, they, too, can be argued about. A Question Influences a Claim if the plausibility of the Claim depends on how the Question is answered. We discuss other objects such as Group and Viewpoint in [Lee 90].

A Session with SIBYL

In this section, we explain the way in which people contribute their knowledge that is used by SIBYL to construct a knowledge base represented by a decision graph, such as shown in Figure 2. Most of the features we describe below have been implemented on top of Object Lens [Lai et. al. 89], a general tool for building computer supported cooperative work applications. SIBYL has been used for several real-life decision making cases such as choosing the optimal hardware platform for a project and cooperatively designing a floor space. We plan to report on these experiences in another paper.

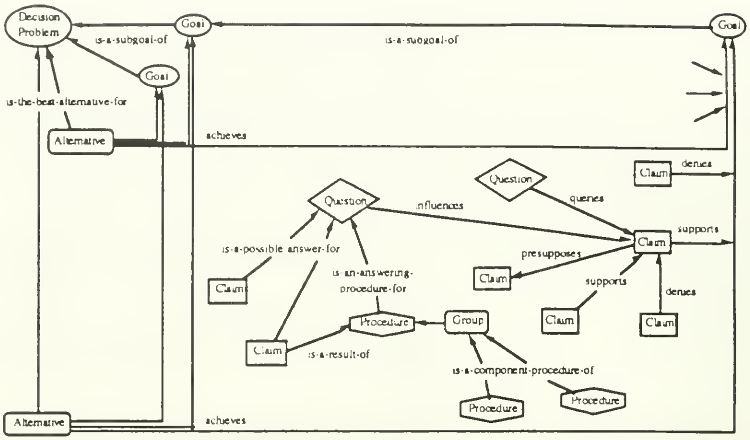


Figure 1. DRL Vocabulary

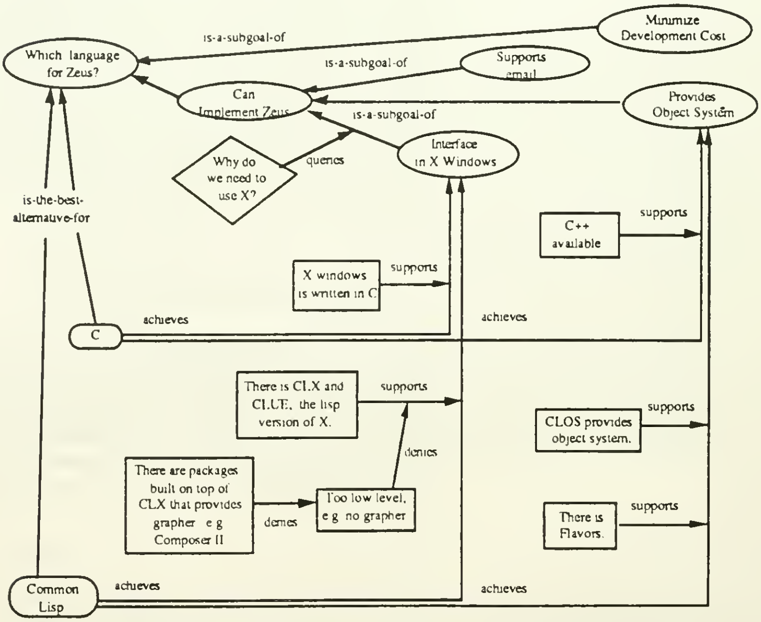


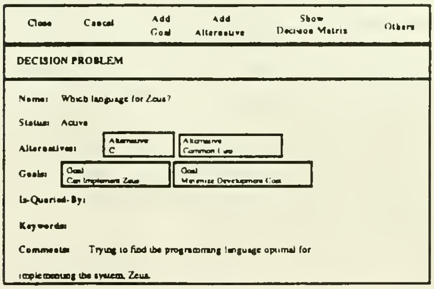
Figure 2. An Example Decision Graph

The Problem

Imagine a group that is trying to decide which programming language to use for implementing a system called Zeus. Jane, the group manager, starts the decision process by creating an instance of Decision Problem. She does so by mouse-clicking on the type object, Decision Problem, displayed in the type browser showing all the types as a lattice. That brings up a template editor displaying the new instance as well as a set of menu items representing the actions that can be legally performed on the object . Figure 3 shows such an editor except that some of the fields have been edited by the user in the manner we describe below.

Specifying Initial Goals and Alternatives

Jane mouse-clicks on the option, Add An Alternative, to create Alternative objects standing for the alternatives being considered, C and Common Lisp in our example. Similarly, through the action, Add A Goal, Jane creates Goal objects standing for the properties that an optimal alternative should have: "Can Implement Zeus" and "Minimize Development Cost". These goals and the alternatives are automatically associated with the decision problem instance by SIBYL, as shown in Figure 3. The manager then asks for a decision matrix for this decision problem. A Decision Matrix, such as shown in Figure 4, is the major interface between SIBYL and the user. It displays the goals in the top row and the alternatives in the leftmost column. The value in each cell represents the current evaluation of the alternative with respect to the goal associated with the cell. Initially, each cell displays the value, "unevaluated". As people produce pro and con claims for the alternatives, as we describe below, the values of the cells get updated,



DECISION PROBLEM

Name: Which language for Zeus?

Status: Active

Alternatives: Alternative C | Alternative Common Lisp

Goals: Goal Can Implement Zeus | Goal Minimize Development Cost

Is-Queried-By:

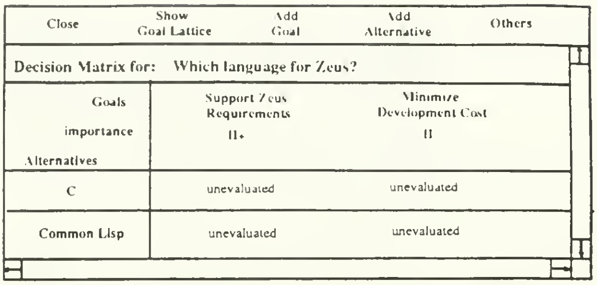
Keywords:

Comments: Trying to find the programming language optimal for implementing the system, Zeus.

Figure 3. A Decision Problem instance

Getting Input from Group Members

Jane announces this decision matrix instance to her group members to solicit their input. We discuss later what kinds of communication SIBYL allows among group members. For simplicity, assume for now that there is a machine running SIBYL and people can walk up to it to add their opinions/knowledge or examine those that have been added since the last time. The group members contribute to the decision making by evaluating the alternatives with respect to the goals, questioning existing evaluations, or pointing out additional or unnecessary goals.



Decision Matrix for: Which language for Zeus?

Goals / importance / Alternatives	Support Zeus Requirements	Minimize Development Cost
importance	H+	H
C	unevaluated	unevaluated
Common Lisp	unevaluated	unevaluated

Figure 4. A Decision Matrix

Elaborating Goals

John, the major architect of Zeus, is the first person to add to the knowledge base. He is a strong believer in Lisp. He believes that Lisp is good for implementing Zeus because Zeus requires an object system, which Lisp provides by having packages such as CLOS and Flavors available. He also believes that the window system for Zeus must be written in X window system. Although he knows that X is written in C, he thinks that Lisp will do fine because there is CLX and CLUE -- Lisp implementations of the X library and toolkit. To represent his support for Lisp, John first examines the decision matrix, shown in Figure 5, and decides to elaborate the goal, "Support Zeus requirements", into subgoals. He mouse-clicks on the cell that says "Support Zeus requirements", chooses from the pop-up menu the item, Add A Subgoal, and creates two additional goals: "Provides Object System" and "Interface in X windows". Similarly, John creates an additional subgoal, "Supports email", because he knows that Zeus requires this feature as well although he does not have specific arguments in favor of either alternative. These subgoals represent John's beliefs about the requirements of implementing Zeus; as such, these beliefs can be argued about or reused, for example, when one has to delineate the requirements of a system similar to Zeus*. SIBYL updates the decision matrix and displays these additional goals.

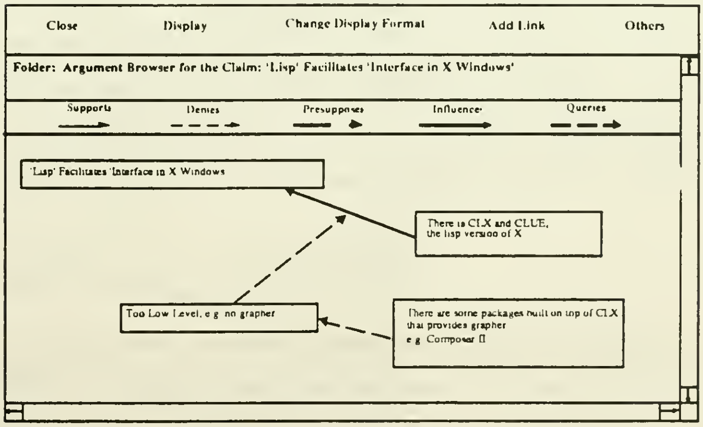
Adding Arguments and Counter-Arguments

---

* If John also believes that these subgoals exhaust the parent goal, in the sense that satisfying them is equivalent to satisfying the parent goal, that they are disjunctive in the sense that satisfying one of them is equivalent to satisfying the parent goal, or that these subgoals are mutually exclusive, or depends on each other, then John can specify these different relations by relating the subgoals to the parent goal via a Group object and specify these relations as one of the attribute of the Group object. This information about the relationship among the subgoals is used by the plausibility management later when the plausibility of the claims are propagated through these Is-A-Subgoal-Of relations. However, John does not specify any relation because he accepts the default relation -- conjunctive, independent, and non-exhausitive. Also, we should note that in DRL Decision Problem is in fact the top level goal of the form, "Choose X for Y", and all the other goals elaborate on this goal. For example, the goal "Can Implement Zeus" should really be interpreted as "Choose X for Y that can Implement Zeus".

John now adds his arguments by mouse-clicking on the cells in the decision matrix that is associated with Common Lisp and the appropriate goals -- namely "Provides Object System " and "Interface in X window" -- and by choosing from the pop-up menu the action item, Show Argument Browser. The argument browser associated with Common Lisp and the goal, "Interface in X window", is shown in Figure 5 except that initially it would contain only the single claim that the alternative achieves the goal associated with the cell, the topmost claim in the browser. As mentioned above, a relation is a subclass of Claim in DRL. In particular, the relation, Achieves that links an alternative A to a Goal G is the claim that A achieves the goal G. This claim is initially neither plausible nor unplausible. This claim is automatically generated by the system for each alternative whenever a goal is created, and an argument browser for each cell in the decision matrix initially contains this claim for people to argue about.

People express their pro and con arguments as claims supporting or denying this Facilitates claim; hence, the plausibility of the claim is updated as people add more supporting or denying claim, and represents the measure of how well the alternative is doing with respect to the associated goal. In SIBYL, the user can always mouse-click on an object and get a menu of all possible operations that can be performed on the object. John mouse-clicks on this initial Achieves claim, and gets a menu displaying possible actions such as Add A Supporting Claim, Add A Denying Claim, Add A Question, and Specify A Presupposition. When the user chooses Add A Supporting Claim, for example, a template editor containing the new instance of Claim is brought up and this new instance is automatically linked to the original claim through an instance of the Supports relation.When John chooses Add A Supporting Claim, SIBYL displays a template editor containing the new claim instance, and links this new claim to the original claim via a Supports relation. Figure 5 shows the argument browser after it has been updated after many people's contribution including John's. An argument browser is in fact a window into the portion of a decision graph such as shown in Figure 2 (i.e. the region bound from below and right by the Achieves link linking the alternative and the goal, and bound from top and right by other Achieves links).



Folder: Argument Browser for the Claim: 'Lisp' Facilitates 'Interface in X Windows'

'Lisp' Facilitates 'Interface in X Windows

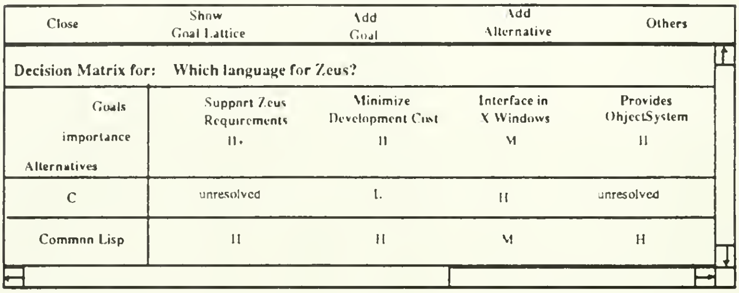
There is CLX and CLUE, the lisp version of X

Too Low Level, e g no grapher

There are some packages built on top of CLX that provides grapher e g Composer II

Figure 5. An Argument Browser

Suppose that several users have contributed their knowledge this way. John comes back and examines the decision matrix. The updated matrix is shown in Figure 6. The value "unresolved" in a decision matrix cell means that there are some issues that need to be resolved before producing the evaluation score for the alternative with respect to the goal associated with the cell. The values, H, M, L stand for High, Medium, and Low, and represent the measure of how well the alternative is doing with respect to the goal associated with the cell, and can be explicitly assigned by the central decision maker (if there is one) or derived from the plausibilities of the pro and con claims as we describe briefly in the next section. He asks SIBYL to show all the claims that have been added since the last time he used SIBYL. SIBYL can display in different formats such as in a table or in a network; or sort them by their creators, by their creation dates, by the claims they are related to, in fact by any one of the fields they have. After examining the existing knowledge base this way, John decides to answer any questions that have been raised in the meantime, respond to any claim that has been advanced so far, or add a new piece of new information that he acquired so far. The resulting decision graph depicting the knowledge base is the one shown in Figure 2.



Decision Matrix for: Which language for Zeus?

Goals / importance / Alternatives	Support Zeus Requirements	Minimize Development Cost	Interface in X Windows	Provides ObjectSystem
importance	H+	H	M	H
C	unresolved	L	H	unresolved
Common Lisp	H	H	M	H

Figure 6. The updated Decision Matrix

SIBYL can run on multiple workstations to allow cooperative, distributed decision making. We have experimented with two kinds of communication among its users: sharing through a special type of messages and sharing through files [Tarazi 89]. Perhaps, it suffices to say here that they are only temporary solutions because the consistency of the knowledge base across multiple sites depend on the order in which messages or files are read. This problem will go away when we use a database with concurrency control as the object server, as we plan to in the near future.

SIBYL Services

Using a decision graph, such as that shown in Figure 2, SIBYL can provide the following four kinds of services: the management of dependencies, plausibilities, viewpoints, and precedents. These services are currently being implemented, and more details can be found in [Lee 90].

Dependency management is responsible for maintaining a consistent state of the knowledge base when a change is introduced. In decision making, objects or their attribute values depend on other objects. For example, in Figure 2, "Interface in X Windows" would not be a goal if the hardware platform does not support X window. SIBYL provides a language in which the user can represent such a dependency, and execute the user-written script when additional information becomes available. A typical action that one can specify is that of updating the plausibility of the claim influenced. Other kinds of actions that can be performed include creating new objects and linking them to existing objects via specified relations.

Plausibility Management is a special case of Dependency Management. It is responsible for maintaining the consistency among the plausibilities of related claims. The plausibility of a claim is partly a function of the plausibilities of the claims related to it. Hence, to compute the plausibility of a claim, we need to propagate plausibilities across the different relations and merge them. The plausibility manager provides an interface for existing schemes for propagating and merging uncertainties such as Bayes' [Duda et. al. 79], Dempster-Shafer's [Shafer 76], or Quinlan's [1983]. This way, SIBYL need not force a particular way of dealing with uncertainty while allowing the use of quantitative methods in the context of qualitative representation.

Viewpoint Management is responsible for creating, storing, retrieving, mapping, and merging Viewpoints. A Viewpoint represents an object that represents a collection of objects that share certain assumptions. (that includes at least a decision problem) Multiple Viewpoints on a decision record represent multiple perspectives on the given decision problem . For example, in Figure 2, if we wanted to see the effect of assigning different importance to a given goal, we can create multiple viewpoints that correspond to the different weights (including zero weight) on the goal in question. Viewpoints are first class objects in DRL. As such, they can appear as alternatives in a meta decision problem such as whether we should stop exploring additional alternatives. Also, Viewpoints can be related to one another more than chronologically. The following are some of the useful relations that DRL allows or will allow among viewpoints: Is-A-Next-Version-Of, Elaborates, Restricts, Has-Different-Importance, Has-Different-Plausibilities.

Precedent Management helps knowledge sharing across groups. It is responsible for indexing past decisions and retrieving ones useful for the current decision problem. Once they are retrieved, it has the job of extracting from them the pieces of the knowledge that are actually relevant for the present problem and placing them in the present decision graph. Sibyl uses goals to index past decisions. Two decision problems are judged to be similar, thus potentially useful, to the extent that they share goals. Goals, in turn, are judged to be shared if they belong together in the potential matches that SIBYL generates and the match is confirmed by the user. At the present, SIBYL generates potential matches by using its goal tree: all instances of the same goal type are regarded as potential matches. A goal tree starts with a few generic types, such as Minimize Cost; the user creates a goal by instantiating a type from this tree. Users add subtypes as needed, resulting in the incremental growth of the goal tree. Using goals as the index also allows the system to determine which parts of the retrieved decision graph are actually relevant. It is those objects -- the claims and the alternatives -- that are linked to the shared goals and their subgoals. We can, so to speak, lift out the shared goals and the subgoals and take with them all the objects that are linked to them. We place this structure in the current decision graph by overlaying the shared goals. This way, a decision maker becomes aware of more alternatives or different ways of achieving a given goal.

RELATED WORK

Recently, there have been a few studies with the similar aim of representing decision rationale or arguments. ArgNoter [Stefik et. al. 87], although one of the earliest, is closest in spirit to SIBYL; Both try to facilitate group decision making by making the

structure of argumentation explicit and helping to manage the dependencies in the structure. Unfortunately, ArgNoter remains a set of high level descriptions; for example, it is not clear what representational structure should be used to realize its features. SIBYL can be viewed as an attempt to articulate further the ideas behind ArgNoter and make them concrete. SYNVIEW [Lowe '86] is one of the earliest attempts to represent arguments among the users distributed over a network. However, for that reason, its representation is severely limited, namely that of indented text with indentation sometimes meaning alternative, other times meaning evidence relations. SYNVIEW is also interesting for its attempt to provide some measure of consensus, which is an important topic of research in group decision making.

[Marshall 87] is interesting because it suggests the use of the matrix structure, similar to the decision matrix of SIBYL, not only to evaluate the different alternatives with respect to the different goals, but also inversely to infer the goals of an agent based on the decisions it made. Marshall's representation is similar to DRL in that it associates with each cell a claim structure, where a claim can be related to other claims through Evidence or Assumption link, corresponding to Support/Deny and Presupposes relations of DRL. However, it does not provide a way to express how the goals are related among themselves; nor does it allow the evidence or the assumption to be argued about. Hence, DRL can be viewed as an elaboration of the structure of Marshall's. MacLean et. al. [89] describes the benefits of explicitly representing design rationale, but as they point out, their aim is to represent the logical space of design alternatives, i.e. systematic representation of the possible alternatives and their evaluations, rather than the actual deliberation process, as SIBYL tries to do. Nevertheless, these two aspects can be quite complementary and we plan to study how they should be related.

Potts and Bruns [88] proposes that a design history be captured as an alternation of design artifact and a set of rationale, each of which is in turn captured by the IBIS structure [Kunz and Rittel 70]; they use this structure to represent a very interesting example of software design . The DRL structure of SIBYL can be viewed as an alternate structure for representing rationale. Below we compare SIBYL to gIBIS, another IBIS-based system, and discuss at length why we believe the DRL structure is more useful than the IBIS structure for representing rationale. The design environment of [Fischer et. al. 89] also aims at recording design rationale and providing useful services in the domain of kitchen design. It is different from SIBYL in that the representation used for design rationale is again IBIS-based, and the service that it provides are mainly that of suggestions and critique. Also, by restricting the domain, it can exploit the domain-specific knowledge, though as a consequence it becomes less general.

gIBIS (graphical Issue Based Information System) is a "hypertext tool for exploratory policy discussion" [Conklin & Begeman 88]. The goal of gIBIS is to capture the design rationale: "the design problems, alternative resolutions, tradeoff analysis among these alternatives, and the tentative and firm commitments that were made in the process of the decision making process". In the rest of this section, we compare SIBYL to gIBIS in detail because gIBIS is well-known, its goal is quite similar to SIBYL's, and the IBIS structure that gIBIS uses is adopted by many other systems, as mentioned above. Figure 7 shows the objects and relations that form the language of gIBIS. SIBYL shares with gIBIS the goal of representing the knowledge that accumulates in the process of design or decision making so as to make it available for review or reuse. Both SIBYL and gIBIS achieve this goal by providing a language whose vocabulary consists of objects and relations generic to a certain type of task, namely decisions which involve evaluating alternatives through arguments.

We view SIBYL, however, as an extension of gIBIS. Elsewhere [Malone et. al. 89], we considered a continuum between a simple hypertext system which has only syntactic types (e.g. text node, image node) and a full-fledged knowledge base where all the knowledge is represented formally for automatic processing by the system, and we argued for systems with middle grounds. Both gIBIS and SIBYL take these middle grounds, but SIBYL is closer to the knowledge base end than gIBIS. gIBIS is mainly a hypertext system whose services focus on the presentation of structure with the purpose of making it easy for people to see the structure of what is represented.and enter additional pieces of knowledge. Although gIBIS has the semantic types, as shown in Figure 7, these types are used mainly for the purpose of presentation: enabling people to see clearly the structure of the represented knowledge, navigate through semantic links, and enter additional pieces of knowledge at appropriate places. On the other hand, we view SIBYL mainly as a knowledge-based system which uses semi-formal representation so as not to force people to formalize all the knowledge that they contribute. The services that SIBYL provides are, for example, typical of knowledge-based systems: management of dependency, uncertainty, viewpoints, and precedents.

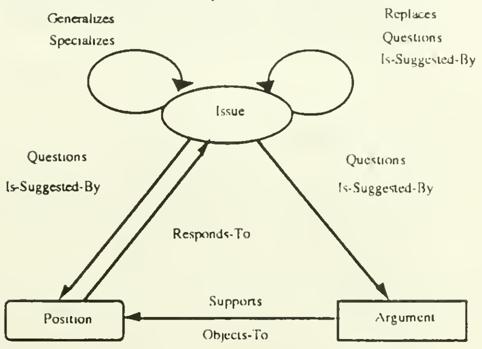


Figure 7. The gIBIS Vocabulary

The difference between SIBYL and gIBIS in the services that they aim to provide might explain the difference in the vocabulary between the two. Issue in gIBIS corresponds to Decision Problem in DRL, Position to Alternative, and Argument to Claim. Notably lacking in gIBIS, however, is the notion of goal or objective against which the alternatives are being evaluated. In gIBIS, objectives are only implicitly represented. For example, one can support the position, "Use Lisp", with the argument "Lisp provides an Object System because there is CLOS". From this relation, one can probably infer that providing an object system is an objective, but it is not clear why it is an objective or how it is related to other objectives. Worse, often intermediate objectives are not mentioned at all in an argument. Consider, for example, another argument "C has C++" supporting the position, "Use C". In this case, the objective of having an object system is only implicit, hence more difficult to argue about. Also, without the explicit representation of the goal, "Provides Object System", it is not clear that the above two arguments -- one mentioning CLOS and the other mentioning C++ -- are comparing the alternatives on the same attribute. These reasons may, at least partially, explain the difficulty of gIBIS in helping people come to a consensus [Conklin and Begeman 88].

Thus, there are many good reasons for explicit representation of goals. It makes people articulate and become aware of the objectives against which alternatives are being evaluated. The explicit representation also allows people to argue about these objectives and change them if desirable. For example, if a goal were to change, say if "Providing Object System" was no longer an important subgoal of the goal, "Can Implement Zeus", then in SIBYL one needs only to delete the subgoal (or set its importance to zero), which will automatically nullify all the claims associated with it., whereas in gIBIS there is no modular way of accommodating this change.

Goals also play important roles in all of the SIBYL services. The explicit representation of goals also allows the dependency manager to update the knowledge base when goals change: for example, Pascal appears as an alternative only if the importance of the goal of making the system educational becomes high enough. The plausibility of an Is-the-Best-Alternative-For relation is a function of the importance of the goals and the plausibility of the Achieves relations linking an alternative to each of the goals. In gIBIS, we cannot talk about what the objectives are, let alone their different importance. As we discussed, one often wants to create multiple viewpoints based on which goals are considered important. And we discussed how the precedent manager used goals as a basis for determining which pieces of knowledge from past decisions are useful to the current decision. There are other structural differences between SIBYL and gIBIS, such as whether a relation is the first-class object that can be argued about. For example, in gIBIS, one cannot say, "I agree with A and B but not that A supports B" or "I don't think G should be a goal". Overall, it seems to us that the structure of DRL is more expressive and motivated by the services that SIBYL provides, whereas the gIBIS structure is motivated mainly by easier human use.

CONCLUSION

We pointed out the benefits of explicitly representing the knowledge gathered in a decision making process, and described SIBYL, which provides a language for describing such knowledge and a set of services that reward the use of the language. We then compared SIBYL to gIBIS, which has the similar goal of representing design rationale, and argued that SIBYL extends gIBIS ~~SIBYL~~ by providing more services. Are more services always better? To what extent should SIBYL try to incorporate other decision support capabilities such as computing expected utility or easier on-line database access, provide interfaces to them, or ignore them? We conclude the paper with a design principle that we have used to answer the above question in developing SIBYL -- the principle that was in turn derived from our experience with developing SIBYL itself.

We believe that a system which needs to elicit knowledge from people, such as SIBYL, should strive for as many services as possible as long as the structure these services require is still simple and natural for people to use. Consider the following heuristic based on this principle: First, examine all the services that we want to provide for a task and rank them by their importance. Find common structures that will accommodate as many important services as possible and natural for people to use. Choose the structure that roughly maximizes these considerations: the number of services weighted by their importance and the ease of use for people. For those services whose required structures still map to the chosen structure, perhaps with additional efforts, try to provide an interface.

We illustrate this heuristic with two examples. As compared to gIBIS, SIBYL uses only a slightly more complex structure (i.e. type-specific attributes, explicit representation of goals, relations being first class objects), which we believe is still natural to the user, if not even more so. However, this structure of SIBYL, i.e. DRL, has been designed with the services that we discussed in Section 2.3 in mind. As a consequence, SIBYL can provide more benefits to the users with requiring much more efforts from them. On the other hand, there are some services that SIBYL could have provided, but does not because they require using the representation unnatural to people. For example, we could have designed the DRL structure so as to accommodate Pearl's uncertainty management scheme by making the distinction between Causal Support and Evidential Support as well as requiring people to specify the joint probability distributions. However, these requirements demand efforts unnatural to people, and we plan to have SIBYL provide only an interface to this scheme in the following sense. SIBYL will allow the user specify which uncertainty mechanism she wants to use, and only if the specified mechanism is Pearl's, then SIBYL asks the user for the needed information, e.g. whether each of the existing Supports relations is causal or evidential. The DRL structure that we have is probably not yet optimal in the sense that we have just discussed above. However, we plan to find out by experimenting with SIBYL.

ACKNOWLEDGEMENT

The comments from Thomas Malone, Paul Resnick, and Mark Ackerman on the previous versions of this paper as well as the discussion with Kum-Yew Lai, and Patrick Winston helped clarify my thoughts on the subject of this paper. Frank Halasz also helped clarifying the relationship between SIBYL and some other work. I thank them.

REFERENCES

[Conklin and Begeman 88] Conklin, J. and M. L. Begeman "gIBIS: A Hypertext Tool for Exploratory Policy Discussion" ACM Transactions on Office Information Systems, 6(4) pp. 303-331.

[Doyle 80] Doyle, J. **A Model for Deliberation, Action, and Introspection**, AI TR 581, MIT, Cambridge, MA.

[Duda et al. 79] Duda, R. O., P. E. Hart, and N. Nilsson "Subjective Bayesian Methods for Rule-based Inference System," In Webber, B. and N. Nilsson (Eds.) **Readings in Artificial Intelligence**, Tioga Publishing Company: Palo Alto, CA.

[Fischer et. al. 89] Fischer, G., R. McCall, A. Morch. "Design Environments for Constructive and Argumentative Design" Proc. of CHI'89 Austin, TX. pp. 269-276

[Howard and Matheson 81] Howard, R. A. and J. E. Matheson "Influence Diagram" in Howard, R.A. and J.E. Matheson (Eds.) **The Principles and Applications of Decision Analysis** v.2 Strategic Decision Group, Palo Alto, CA.

[Kunz and Rittel 70] Kunz, W. and H. Rittel "Issues as Elements of Information Systems" Working Paper No. 131, Institute of Urban and Regional Development, Univ. of California Berkeley, CA 1970

[Lai et al. 89] Lai, K., T. Malone, and K. Yu "Object Lens: A 'Spreadsheet' for Cooperative Work" ACM Transaction on Office Information Systems, 6(4) pp. 332-353.

[Lee 89a] Lee, J. "Decision Representation Language (DRL) and Its Support Environment", MIT AI Lab, Working Paper no. 325, August.

[Lee 89b] Lee, J. "Task-Embedded Knowledge Acquisition through a Task-Specific Language", Proceedings of IJCAI 89 Workshop on Knowledge Acquisition, Detroit, MI.

[Lee 90] Lee, J. "SIBYL: A Qualitative Decision Management System", to appear as Ch. 5 in Winston, P. with S. Shellard (Eds.) **Artificial Intelligence at MIT: Expanding Frontiers**, The MIT Press, Cambridge, MA.

[Lowe 86] Lowe, D. "SYNVIEW: The Design of a System for Cooperative Structuring of Information," Proceedings of CSCW'86, Austin, TX.

[MacLean et. al. 89] MacLean, A., R. M.Young, T. P. Moran "Design Rationale: The Argument Behind the Artifact" Proc. CHI'89 Austin, TX. pp. 247-252

[Malone et. al. 89] Malone, T., K. Yu, and J. Lee "What Good Are Semi-Structured Objects? Adding Semiformal Structure to Hypertext" Sloan Working Paper # 3064-89-MS, MIT

[Marshall 87] Marshall, C. "Exploring Representation Problems Using Hypertext" Proc. of Hypertext 1987, pp.253-268

[Pearl 88] Pearl, J. **Probabilistic Reasoning in Intelligent Systems**, Morgan Kaufmann: Palo Alto, CA.

[Potts and Bruns 87] Potts, C. and G. Bruns "Recording the Reasons for Design Decisions" MCC Tech Report STP-304-87

[Quinlan 83] Quinlan, J. R. "Inferno: A Cautious Approach to Uncertain Inference," The Computer Journal, 26(3): 255-267.

[Raiffa 68] Raiffa, H. **Decision Analysis**, Addison-Wesley, Reading, MA.

[Shafer 76] Shafer, G. A **Mathematical Theory of Evidence**, Princeton Univ. Press, Princeton, N.J.

[Stefik et. al. 87] Stefik, M., G. Foster, D. Bobrow, K. Kahn, S. Lanning, and L. Suchman "Beyond the Chalkboard: Computer Support for Collaboration and Problem Solving in Meetings", Comm. of ACM, 30(1)

[Tarazi 89] Tarazi, M. H. Object Sharing in a Multi-User Hypertext System, M.A. Thesis, Dept. of EECS, MIT. Cambridge, MA.

[Toulmin 69] Toulmin, S. **The Uses of Argument**, Cambridge Univ. Press, Cambridge, England.

[Yakemovic and Conklin 90] Yakemovic, K.C.B. and J. Conklin "Observations on a Commercial Use of an Issue-Based Information System" a draft submitted to CSCW '90 LA:CA