"INTEGRATING CASE-BASED REASONING IN MULTI-CRITERIA DECISION SUPPORT SYSTEMS"

by

Albert ANGEHRN *
Soumitra DUTTA **

92/54/TM

* Assistant Professor of Information Systems, at INSEAD, Boulevard de Constance, Fontainebleau 77305 Cedex, France.

** Assistant Professor of Information Systems, at INSEAD, Boulevard de Constance, Fontainebleau 77305 Cedex, France.

Printed at INSEAD,
Fontainebleau, France

Integrating Case-Based Reasoning in Multi-Criteria Decision Support Systems

Albert A. Angehrn
Soumitra Dutta

INSEAD, European Institute of Business Administration,
Fontainebleau Cedex, 77305 France

Abstract
An important focus in current research on decision support systems (DSS) is the design of flexible environments to facilitate and support learning about the problem domain by the user. This research uses case-based reasoning to present a *symbiotic* DSS in which both the user and the DSS learn from each other. The user learns from the DSS (from stored prior problem solutions) and the system learns from the user (by observing current problem solving behaviors). The specific context of our research is the class of DSS used for supporting multi-criteria decision making (MCDM). Our ideas are being implemented in a prototype extension of the *Triple C* [2] MCDSS.

1 Multi-Criteria Decision Support Systems

Multi-Criteria Decision Support Systems (MCDSS) are interactive, computer-based systems helping decision-makers (individuals and/or groups) to solve various semi-structured and unstructured problems involving multiple attributes, objectives and goals. A recent survey of MCDSS [9] describes the wide applicability of these systems and indicates their importance within the DSS research field (see also [12], [5] and [23]). Typical problems addressed by MCDSS can be characterized as being composed of four main components: a user (an individual decision-maker or a group), a (not necessarily explicit) preference structure, a dynamic, finite set of alternatives (potential actions), and a set of criteria to be considered in the evaluation of alternatives (see Figure 1).

In order to provide support during the different phases of a decision-making process, the majority of current MCDSS rely on one (or more) of the methodologies developed within the MCDM (Multi Criteria Decision Making) research field (see e.g. [13, 28]). Such methodologies address multi-criteria decision making as a problem which can be mapped onto a (generally static) formal model, and then solved using suitable mathematical techniques (see for instance the *Expert Choice* system based on Saaty's Analytical Hierarchy Process [22], the family of methods *Electre* developed by Roy [21], as well as a wide range of systems based on Multi Attribute Utility Theory [25]). In addition to mathematical models, MCDSS typically include heuristics or rule-based reasoning [15,14], and databases, as illustrated in Figure 1. Mathematical techniques developed in the MCDM field have had a major impact on the design of the early MCDSS. As a result, the majority of current MCDSS are merely *computerized versions* of MCDM techniques (as described above).

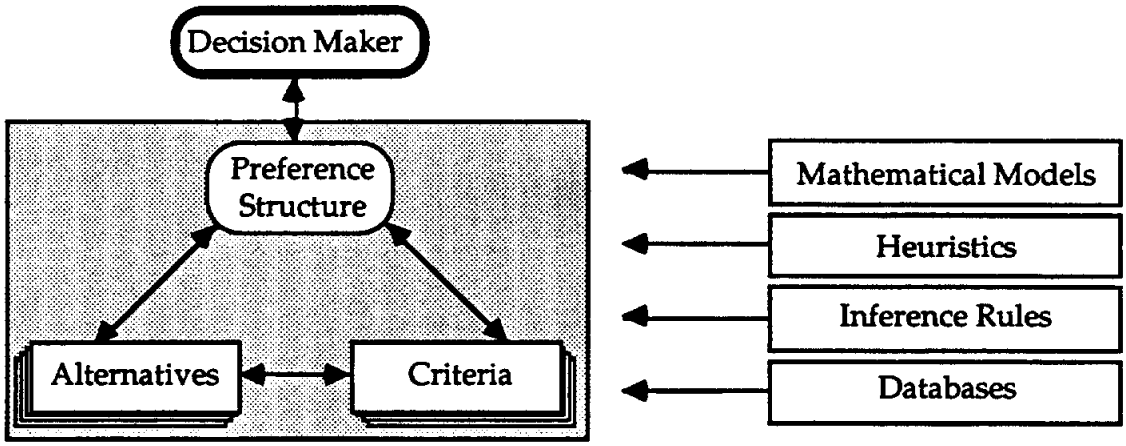

Figure 1: MCDM problem components and MCDSS support tools

More recently, the suitability of computerizing MCDM techniques for the design of MCDSS has been questioned by different researchers (see e.g. [27, 2, 4]). The essence of their arguments is the belief that the role of MCDSS should not be limited to "imposing relatively simple (and rigid) mathematical artifacts on the rich, natural, self-organizing, and knowledge-producing processes of individual and social decision making [27]." They argue that research in the MCDSS area should focus on defining "flexible environments in which *individual learning* about a decision situation can take place [2]," i.e. emphasize interactive systems which support decision makers in recursively redefining their problems and updating their aspirations, until a form of stability (*cognitive equilibrium* [27]) is obtained.

As detailed in Section 3, the *learning* oriented (as opposed to a problem solving) approach advocated in the above paragraph provides a conceptual basis for the design of a new class of user-centered [17], symbiotic [11] MCDSS whose contribution to a decision making process can be measured in terms of facilitation of and stimulus [4] to (reflective) learning and understanding (rather than solely in terms of supported techniques and technology).

2 Case-Based Reasoning

CBR (or analogical reasoning), though common and extremely important in human cognition, has only recently emerged as a major reasoning methodology within Artificial Intelligence (AI) (see [19] for a good survey of CBR). CBR, in general, is concerned with the solution of problems by identifying and adapting similar problems stored in a library of past experiences/solutions. The recent popularity of CBR is due to the realization of the dual facts that (a) it is difficult to obtain rules and update them periodically (knowledge engineering is widely recognized as the critical bottleneck in the building of expert systems), and (b) rule-based systems respond poorly to new situations because they are unable to learn from experience and adapt their prior problem solving procedures.

While rules are difficult to obtain, descriptions of prior problem solving behaviors (i.e., prior cases) are relatively easy to locate. For example, hospitals keep treatment records of patients and engineering design firms keep a store of prior design models. Prior cases are used extensively by humans for problem solving in new situations. A doctor utilizes knowledge about cases treated in the past while diagnosing and prescribing treatment for a new patient. Engineering design teams build on their prior knowledge while designing a new component.

CBR provides a deeper level of knowledge and reasoning as compared to simple rule-based reasoning (RBR). Rules represent the distilled commonalties (also termed as expertise within the literature of expert systems) from many different cases. Rules capture features common to many cases, but they also tend to impose rigid solution structures (a consequence of capturing "commonalties"). Thus simple rule-based systems are generally poor at dealing with exceptions (which are not common to all cases) and new previously unseen situations. In contrast to RBR, CBR does not attempt to find and store commonalties. Rather it stores prior cases (descriptions of previous problem solutions) and when faced with a new problem it solves the problem by finding the most similar (to the current problem) case and adapting the old solution to the current problem. Thus CBR is, in principle, more powerful than RBR in its capabilities to deal with exceptions and new situations. Figure 2 depicts the basic solution structure of CBR.

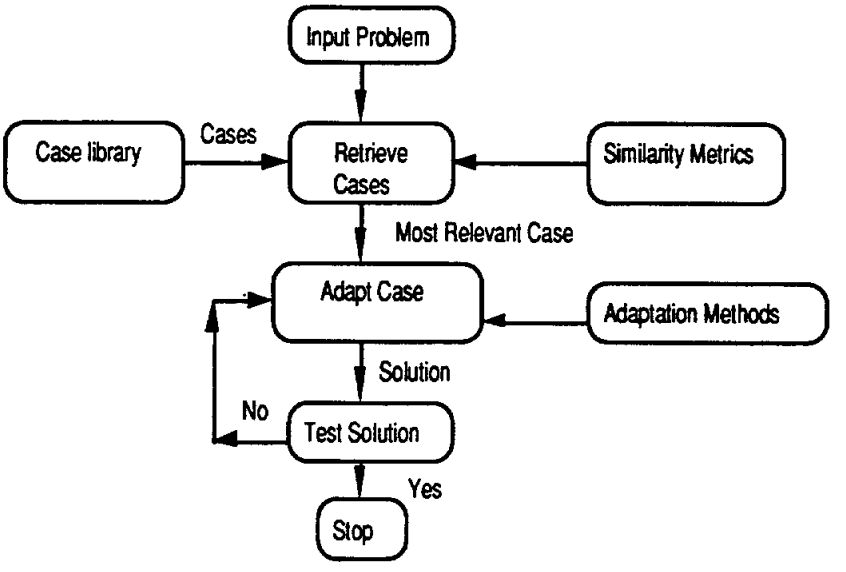

Figure 2: Solution structure of case-based reasoning

CBR, while simple conceptually, is not very easy to implement. There are several important research issues associated with CBR. A case library can contain potentially many hundreds of cases. The organization and retrieval of cases from a large case library is a non-trivial issue. Two key notions in this context are those of *similarity* and *relevancy* [8]. The term similarity refers to the "closeness" of a case to the current problem while the term relevancy refers to the degree of usefulness of a particular case for solving the current problem. CBR is most successful only when the retrieved case is both similar and relevant. Note that mere similarity is not sufficient for CBR. For example, if two persons have the same nationality, then it is very likely that they also have the same native language (as nationality is relevant for determining the native language of a person). However, if two persons have the same height, then it does not mean that they have the same native language (because height is not relevant for determining the native language). Appropriate definitions of similarity and relevancy metrics are important research issues within CBR.

Another important problem is determining when to use CBR. CBR is neither always desired nor is it always useful. Certain protocols have to be devised to formalize the process by which CBR is triggered. Also, CBR in isolation is typically not very useful. It has to be integrated with a knowledge-based component (usually expressed by rules) representing domain knowledge. The integration of CBR with other reasoning methodologies (such as RBR) is recently receiving attention from researchers [7, 20].

3 Integrating CBR in MCDSS: An Overview

Our approach to integrating CBR in MCDSS can be seen in the perspective of an extension of the *decision system* illustrated in Figure 3. Such a decision system consists typically of a human decision maker, and a computer-based system, with both being viewed as resources in the decision making process [24] and assuming different roles throughout the interaction process. For instance, the role played by the human component of the decision system - the user - consists primarily in providing (1) the context for the interaction, i.e. the user's initial problem perception, (2) the motivation for trying to explicitly represent and explore the given decision situation, and (3) the judgment driving this exploratory process. On the other side, the role of the computer-based system - the MCDSS - can be reduced to two main components, as outlined in section 1:

(1) A *Facilitation Component*, i.e. a set of integrated tools enabling decision makers to incrementally map their mental models into explicit representations which can be revised, analyzed and explored.
(2) A *Stimulus Component*, i.e. a set of tools which *actively enhance* learning and understanding throughout the computer-supported decision making process.

In terms of *Facilitation*, a MCDSS should provide a variety of tools which users can employ flexibly to define relevant criteria, describe alternatives and express their preferences in terms of criteria importance, desired alternatives, aspiration levels, ideal values, and so on. In addition, representation tools and feedback mechanisms should support the generation of different problem views, the comparison of alternatives as well as "What If" type of analyses. A concrete example of a fully-fledged Facilitation Component within a MCDSS is illustrated in reference [2], a paper reporting on the main characteristics of a visual interactive system called *Triple C*.

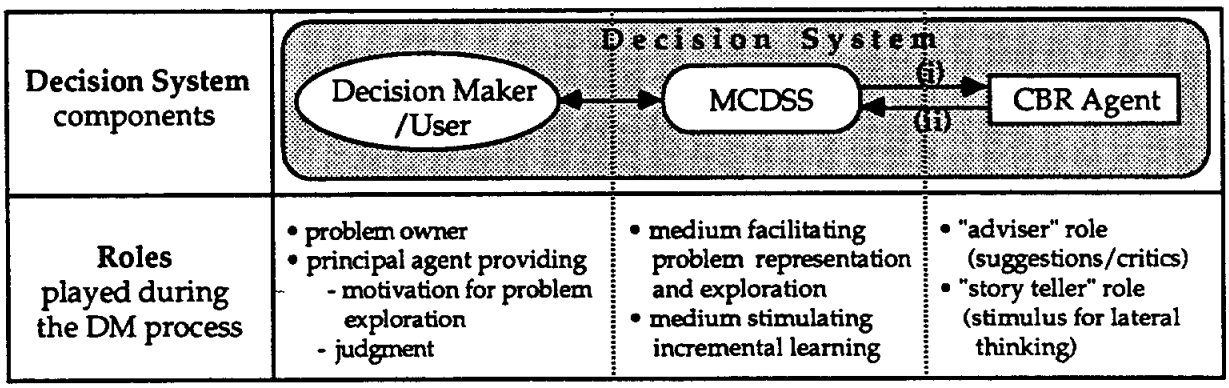

Figure 3: Overview of components and roles in a Decision System.

On the other hand, the design of the *Stimulus Component* of a MCDSS requires the design and integration of tools which *actively* enhance the decision makers' understanding of the modeled situation and ideally prevent interruptions in the learning process due to a premature feeling of satisfaction [27]. As discussed extensively in [4], the integration of so-called *Stimulus Agents* in a DSS is currently one of the major challenges in the field of DSS (cf. also the concepts of "intelligent" and "active" DSS in [16] and [1]). In addition, the concept of *Stimulus Agents* discussed in [4] provides an alternative framework for integrating advanced mathematical and AI techniques within DSS. For instance, the use of optimization methods within the MCDSS *Triple C* [2] aims primarily at stimulating the decision-maker to reflect on the justification of a given ranking of alternatives. In a different context, the knowledge-based component of the DSS *Tolomeo* (see [1, 3]) stimulates the

improvement of user-designed solutions by logically inferring and suggesting alternative solutions. Other examples of system integrating Stimulus Agents can be found in references [10,16,18].

The above considerations provide the framework for our attempt to integrate CBR in a MCDSS. In this approach, CBR techniques are encapsulated into an additional Stimulus Agent (called in the following: ***CBR agent***) contributing to the dynamics of the decision system (see Figures 3 and 4) by assuming two different roles: as an *adviser* and as a *story teller*. In the advisory role, the CBR agent can recognize whether a multi-criteria approach was used successfully in other similar decision situations before, and provide information about the types of alternatives, criteria and preference structures used. While not all aspects of a prior situation will typically carry over to the new problem, significant guidance and help can be provided to the user in dealing with a new situation. In the story teller role, the CBR agent can replay the process of obtaining solutions to prior similar MCDM problems. By replaying the solution process of a previously solved problem, the user can gain insights into both how to structure the problem and about the process to obtain a solution. This replay can guide users in tackling new problems or serve as a tutoring aid to inexperienced users.

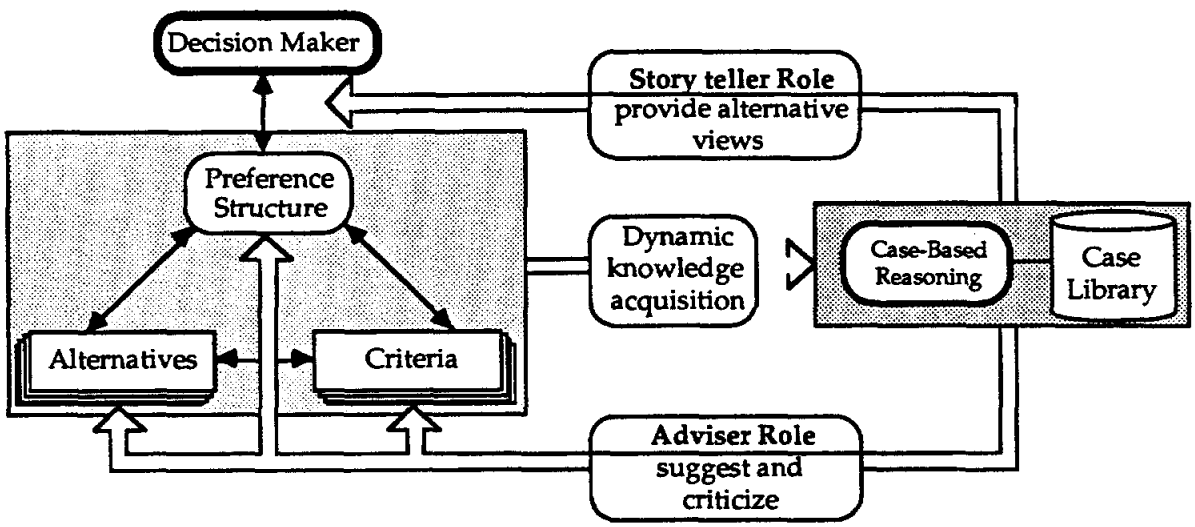

Figure 4: Roles of a CBR agent in the MCDM context

The CBR agent has two major components: a case library (CL) and inference strategies. The CL contains different cases (solutions to prior problems) and is updated dynamically every time a different user employs the system or a different problem is explored. With the help of the CL, the CBR agent maintains a repository for the cumulative problem solving experience of users of the MCDSS. Each case in the CL represents the major components of a multi-criteria decision (alternatives, criteria and preference structures) generated by a particular user while solving a specific problem. The inference strategies of the CBR agent help to determine the conditions which require the use of CBR (see section 5), retrieve the most similar case from the CL, and facilitate the adaptation of the retrieved case to the current problem.

To summarize, the integration of a CBR module into a MCDSS can be viewed as the extension of the man-machine decision system described at the beginning of this section with an additional *Stimulus Agent*. The next two sections report in more detail on the two main characteristics of such an agent: its internal structure (and functioning mechanisms) and its behavior during the process of solving the MCDM problem.

4 Structure of the CBR agent

This section provides a more formal description of important aspects of the integration of CBR and MCDM in our research.

4.1 Representation

Each case in the CL can be initially represented by the triplet (U, P, P_c), where U is a particular decision maker, P is the MCDM problem to be solved, and P_c represents the relevant problem characteristics. There are different ways to define the components of the above triplet. For the purposes of this paper we define U as being characterized by the tuple (N_u, S_u), where N_u is the name of the user, and S_u is the skill level (novice, intermediate or expert) of the user. P is characterized by the tuple (N_p, D_p) where N_p gives the name of the particular problem and D_p specifies the domain of applicability (such as electronics, automotive, and so on) of the case. Note that all problems solved by the MCDSS are of the same type (namely MCDM problems), but belong to different domains. Further, we assume that P_c can be represented by the sextuplet, (A, C, V_c, D_{ac}, W_c, I_c) where:

A is a set of n alternatives $\{A_1, A_2,....,A_n\}$,

C is a set of m criteria $\{C_1, C_2,....,C_m\}$ (against which alternatives are evaluated),

V_c is a 1xm vector such that V_i $(0 < i < m+1)$ specifies the allowable values for criteria C_i. We assume that each such V_i can be specified by either a compact interval on the real axis or as a set of discrete, ordered values (such as: *low, moderate, high*)

D_{ac} is a nxm matrix, with d_{ij} giving the data value for alternative A_i with respect to criteria C_j

W_c is a 1Xm vector such that W_i $(0 < i < m+1)$ gives the subjective weight associated with criterion C_i by the user U. We assume that the weights are order sensitive, i.e., $W_i > W_j$ implies that C_i is more important than C_j to the user, U, in the current problem, P. Further all weights sum to 1, i.e.,

$$\sum_{i=1}^{m} W_i = 1$$

I_c is a 1xm vector such that I_i $(0 < i < m+1)$ gives the specified ideal value for any alternative for criterion C_i.

Note that W_c and I_c together specify the user's preference structure for the particular MCDM problem under consideration. The above six primary components of P_c describing a generic MCDM problem are illustrated in Figure 5. Typically a decision maker using a MCDSS begins with a certain initial formulation of the particular MCDM problem to be solved, P_c^i, and then gradually arrives at the final solution, P_c^f, by passing through a series of intermediate stages (as shown in Figure 6). These intermediate stages may involve changes in one or all of the above six MCDM problem components (P_c) (e.g., alternatives and criteria may be added/deleted, and the preference structure changed). As mentioned earlier (section 3), the MCDSS plays an important role in helping the user in changing these component, and in stimulating the user to engage in certain modes of problem solving behaviors.

In our research each case should capture the progressive evolution of the triplet (U, P, P_c) as the user solves the MCDM problem (Figure 6). Thus each case in our case library actually consists of the union of P_c^i, all intermediate stages, and the final solution P_c^f. Because U and P do not change during the solution of any one MCDM problem, we can represent a case in our representation as:

$$(U, P, P_c^i, P_c^{i+1},...,P_c^f)$$

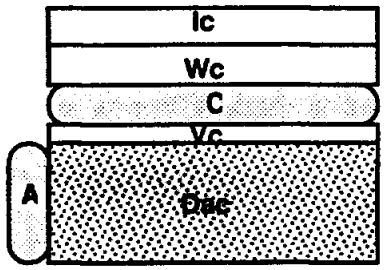

Figure 5: Major components of a generic MCDM problem

We consider any change (made by the user) in the following P_C components: A, C, W_C, and I_C, to lead from one intermediate stage, P_C^{i+k}, to the next stage, P_C^{i+k+1}. Such changes are recorded by the system while storing current problem solutions as cases for future use. Note that our choice of case representation includes process knowledge, i.e., knowledge about how the user started with the initial problem description, and moved through successive intermediate stages to arrive at the final solution. Including such knowledge about the problem solution process into our case representation is necessary to allow the CBR agent to play an effective active role in supporting the decision processes of the user. Note that including such process knowledge into case representations is similar in approach to systems using derivational analogy [6]. However, while derivational analogy considers the process by which an automated planner derives a solution, the process dimension in our approach focuses on providing enhanced on-line support available to a user (via facilitation and stimulus) during the decision making process.

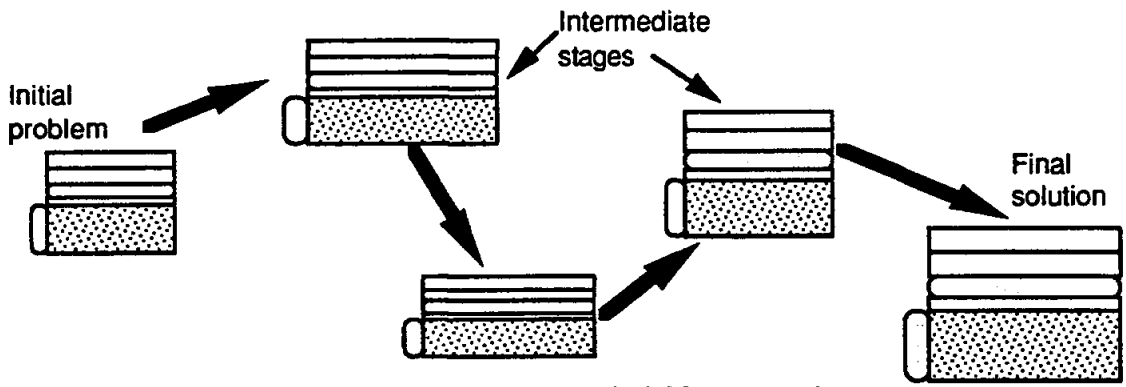

Figure 6: Solution process in MCDM problems

4.2 Structure of the Case Library

The number of cases in a case library is typically very large. A flat storage of cases would lead to inefficient sequential searches. Therefore most systems using CBR employ efficient indexing schemes to store and retrieve cases.

Figure 7 illustrates the structure of the case library adopted in our research. For the case representation described in the previous sub-section, note that P_C (Figure 5) can be adequately represented by a matrix like data structure. Thus the process of solving a particular MCDM problem can be represented by a succession of matrices for the P_C components of each of the intermediate stages. Each long rectangular box in Figure 7 represents a complex matrix-like structure. Successive stages in the evolution of P_C during the solution of any one problem are represented by appropriately labelled boxes. Each case also contains the U and P components, and has a particular identifier (represented simply by a number in Figure 7, but can also contain other valid information such as date and time of creation).

For efficient storage and retrieval of cases, the case library also contains two hierarchical indexing structures (shown partially in Figure 7). The first index structure is used to index the cases on the basis of problem domains. For example, case #1 deals with cars and case #90 deals with the selection of jeeps. The level of granularity of the terms used to index the problem domains is dependent upon the system designer (it has to be decided in conjunction with the user). Similarly, another index structure is used to store and retrieve cases on the basis of the user characteristics. For example, case #1 is solved by an expert user while case #90 represents the solution process of a novice. Special pointer structures are maintained for rapid access to P_c^i and P_c^f, and to the intermediate stages as shown in Figure 7.

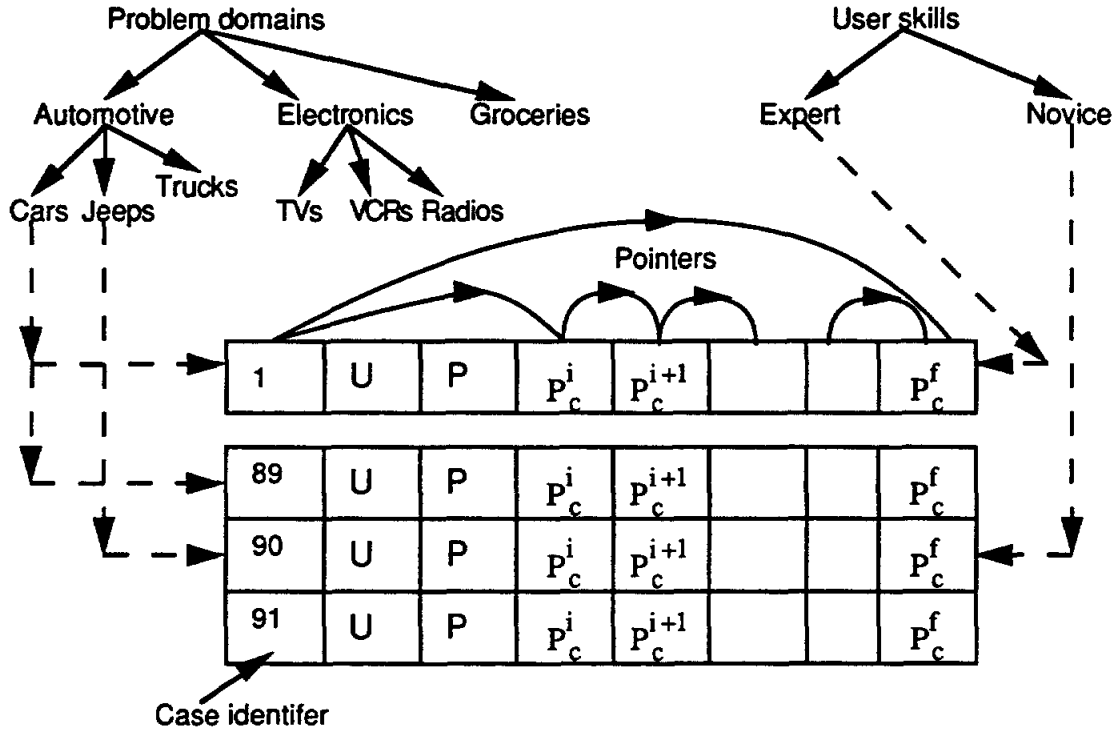

Figure 7: Structure of case library

In addition to the structures shown in Figure 7, the case library also contains a hierarchy of relevancy matrices. The relevancy matrices specify the degree to which a particular domain is relevant for problem solution in another domain. The relevancy matrices are structured in a hierarchy to represent the relevancy measures among siblings at a particular level in the problem domain hierarchy. Figure 8 represents a partial hierarchy of relevancy matrices for the problem domain hierarchy of Figure 7. The value a_{ij} in any relevancy matrix specifies the degree to which domain i is relevant for problem solution in domain j. Note that except for i=j, the value of a_{ij} and a_{ji} need not be equal. The values of the various a_{ij}'s are solicited from the user whenever a new problem domain is added to the domain hierarchy. The hierarchy of relevancy matrices plays an important part in the retrieval of cases as explained in the following sub-section.

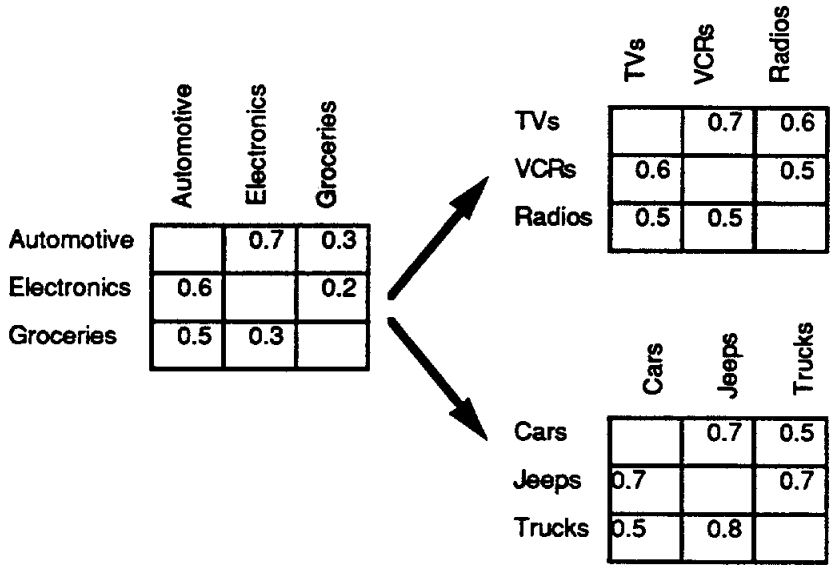

Figure 8: A partial hierarchy of relevancy matrices

4.3 Retrieval and Similarity Metrics

As mentioned in section 3, the CBR agent plays two important roles: as an adviser and as a story-teller. The retrieval mechanisms for each of these roles are explained below.

Retrieval for the story teller role is based on D_p, or U_p or more typically, a combination of D_p and U_p. The retrieval algorithm for such CBR support can be specified in terms of the following steps:

(1) Obtain user's desired index specified in terms of D_p or U_p or a combination of both D_p and U_p. An example of such a request could be "get me a case solved by an *expert* in the domain of *cars*".

(2) Traverse down the domain hierarchy and retrieve a case as determined by the index of step (1). For example, assuming the case library of Figure 7, the example index given in step (1) would retrieve case #1.

The retrieved case in the above procedure is the complete case $(U,P,P_c^i,P_c^{i+1},...,P_c^f)$. The system replays the entire case to the user. There are some complications which may arise in the above simple two-step algorithm:

- The user may not be able to specify a leaf label (such as *car* and *jeep*) in the domain and user skill levels hierarchy. In such a case, the system displays the domain and user skills hierarchy to the user, and lets the user suggest which domain labels are most appropriate.

- There may be more than one case in step (2) which meets the user specified index terms. In such a case, the system can display case identification information along with P and U descriptions of all selected cases, and asks the user to select one or more of the available choices. For example, a user might want to view prior cases solved by a particular expert only (this information is available from the U component of the case).

- There may be no available cases in the leaf labels indexed in step (2). In such a case, the system uses the entries in the appropriate relevancy matrix to determine the sibling node whose solutions are most relevant to that desired by the user. This information is then

conveyed to the user with an explanation of why a solution from another sibling node was chosen for display. For example if the user asked for a case regarding the selection of trucks for the case library of Figure 7, the system would fail to find any prior cases stored under the label trucks. Then the system would access the relevancy matrix (Figure 8) and notice that the domain jeeps is most relevant for the domain trucks, and so it would retrieve case #90. If there are no solutions in any relevant sibling node, the system moves one level up in the domain hierarchy and attempts to find a relevant solution by using the relevancy matrix at the higher level.

The advisory support of the CBR agent is provided when the user needs help (see section 5 for details on how and when this need for help is determined) in structuring specific aspects of P_c. As the user has already started solving a particular problem, the location of this problem in the domain and user hierarchies is known. The system retrieves only the final solution P_c^f in this case and presents it to the user. The user can use P_c^f to guide the modification of either the set of alternatives, or criteria or preference structure. The pointer structures of Figure 7 (note the direct pointer to P_c^f) facilitates the efficient access of P_c^f. Some complications which can arise in the above retrieval procedure are:

- There may be several prior cases in the leaf label under which the current problem is also indexed. For example in Figure 7, two cases #1 and #89, are indexed under *cars*. If the current problem (which is being solved by the user) is also indexed under *cars*, then the system has to decide which case (#1 or #89) to retrieve. Both cases #1 and #89 are relevant for the current problem (as they have the same index terms). If the number of prior cases with the same index terms is small, the system can display information (U, P and identifiers) about all cases or ask the user to select amongst the possible choices. In the case of a large number of prior cases, our system selects a case by comparing similarities between the set of alternatives and criteria in the current problem (as defined by the user so far) and the set of similarly indexed cases. The case with the highest degree of similarity is presented to the user (with an appropriate explanation).

- There may be no prior case for the index terms applicable to the current problem. In such a case, the system uses the relevancy matrices to determine a suitable relevant case, first at the same level in the domain hierarchy, and if that fails, at successively higher levels in the domain hierarchy. The process is similar to that described earlier. Suitable explanations of the process are provided to the user for any retrieved case.

5 CBR Agent's Behavior and Contribution to the DM process

This section describes how the CBR agent interacts with the MCDSS (and thus indirectly with the decision maker). The major interactions between the CBR agent and other components of the decision system (see Figures 3 and 4) can be summarized as:

(i) Dynamic acquisition of problem solving knowledge to be stored in the Case Library,

(ii) Interactive support of the user's problem structuring and incremental learning process through suggestions and critics (*adviser* role) and presentation of alternative problem views (*story teller* role).

5.1 Dynamic Extension of the Case Library

The capability of the CBR agent to collect information from different users and to store it in its Case Library (CL) is important because it directly affects the agent's performance as an

adviser or as a story teller during an interactive session. The three mechanisms allowing the CBR agent to collect, organize and store additional information are:

- A *Monitoring Module* through which the agent records all the events occurring in a session.
- An *Analysis Module* used to translate session protocols into new cases to be stored in the CL or to update existing ones.
- A *Problem Browser Module* supporting the CBR agent in acquiring and incrementally updating information related to (i) problem domains, (ii) user profiles, and (iii) user perceptions of mutual relevance between different problem domains.

The first two modules are directly responsible for a continuous extension and update of the CL. The Monitoring Module provides a way for collecting both static and dynamic information on how a specific user U approached a MCDM problem P by recording all the events caused by the decision maker during his/her interaction with the MCDSS. Such events correspond to changes in one or more of the problem components (formally represented as P_C), i.e. for instance to events such as the introduction of new criteria or to the revision of the user's preference structure. The Analysis Module then reduces a session protocol to a set of relevant state transitions and stores them as a new case in the CL in the form of:

(1) a *final* triplet $(U, P, P_C)_f$, and

(2) a sequence of state transitions of the type $(U, P, P_C)_i \Rightarrow (U, P, P_C)_{i+1}$ reflecting the path traversed by the decision maker before reaching the final triplet.

As described in section 4, the collection of such cases provides a basis for the similarity-triggered matching process underlying the advisory role assumed by the CBR agent during a session. The monitoring and analysis modules also provide the means for the CBR agent to continuously learn and update its own knowledge (i.e., the cases in the CL). These modules in our current system do not support the recording of *causal explanations* associated with transitions between different problem solving stages ("Why, for instance, did user U introduce a new criterion, give low importance to a specific criterion or prefer one alternative to another?"). As causal explanations of state transitions can be an important component of the suggestions/criticisms made by the CBR agent, we plan to integrate it in our system later. Such an extension will require the CBR agent to sporadically interrupt the DM process with requests of information such as: "Why did you just assign so much importance to this specific criterion?" or "Why do you prefer alternative a1 to alternative a2?" Alternatively, the CBR agent will need a knowledge based component which can observe the user's decision process and attempt to use domain specific knowledge to provide causal explanations.

The *Problem Browser Tool* (see detailed description in Appendix 2) has two major functions:

(1) It extends the support provided by the MCDSS by allowing the user to flexibly browse in the Case Library by navigating in the hypertext environment described in Appendix 2.

(2) It supports the CBR agent in collecting information related to (i) the profile of new users (position in the user hierarchy), (ii) the problem domains they address (position in the domain hierarchy), and (iii) the users' subjective judgment of the *relevance* of the problem to other domains (entries in the relevancy matrices - Figure 8).

The *Problem Browser Tool* (PBT) is activated at the beginning of each new session, i.e. whenever a new user employs the system or whenever a new decision situation is explored with the help of the MCDSS. The role of the PBT is that of a *pre-decision support system* confronting the user with different cases, some of which can be considered relevant by the decision maker. If after this pre-decision stage, the user decides to activate the MCDSS in order to map and explore her own problem, the PBT requires:

(1) Information about the user (through a free-text statement about their own background and prior experience with similar cases and a structured self-evaluation in terms of their skill level, e.g., "expert" or "novice").
(2) Information about the problem domain addressed by the user (through a free-text description of the decision situation at hand and an indication of the appropriate position of the problem within the domain hierarchy).
(3) If the new problem has caused a new entry in the domain hierarchy of the CBR agent, a subjective assessment of the relevance of the current problem domain for solving problems in other sibling problem domains (see Section 4 and Figure 8) and vice versa.

Thus the PBT serves as an interactive knowledge acquisition tool and supports the incremental structuring and building of the case library within the CBR agent.

5.2 Agent's Behavior and Interaction with the Decision Maker

The contribution of the CBR agent to the DM process consists of assuming - during a session - the roles of an adviser and a story teller, and stimulating the decision maker in three different ways:

(1) By formulating problem-related suggestions such as "Why don't you consider *experience* as a relevant criterion in your decision?"
(2) By providing constructive criticism such as "Unless you have very good reasons to do so, you should not give so much importance to a criterion like *speed*"
(3) By stimulating lateral thinking through the report of (case-based) stories of how decision makers approached problems in domains different from the one currently explored by the user.

The following paragraphs illustrate the specific manner in which the CBR agent provides these three types of support to the decision maker. In particular, we describe the agent's behavior in terms of the interaction language it employs and its frequency of intervention, i.e. in terms of how and when the CBR agent fulfils its two roles of an adviser and a story teller.

The CBR agent communicates with the decision maker:

(1) Through the manipulation of the different elements and representations of the MCDSS interface (see Appendix 1 for an overview of the interface of the *Triple C* system which is fully documented in [2]).

(2) Through an additional window ("CBR" window) displayed in Figure 9.

The CBR window is used by the decision maker to control ("Give Advice", "Tell a Story", etc.), answer ("Yes", "No"), or request information from the CBR agent ("Why?"). In addition, the window provides the medium through which the CBR agent expresses its recommendations (suggestions, criticisms and story comments) in textual form. To generate such textual recommendations the system employs a predefined set of templates

such as "Why don't you consider *VAR1* as a relevant criterion?" or "Why did you set the maximum value which can be assumed by criterion *VAR1* to *VAR2*? It could be set to *VAR3*." The whole interaction between the agent and the user is mediated through the CBR window. For instance, after receiving a recommendation, the user can request an explanation ("Why?" button) and have the system display details of the case from which the specific suggestion was derived.

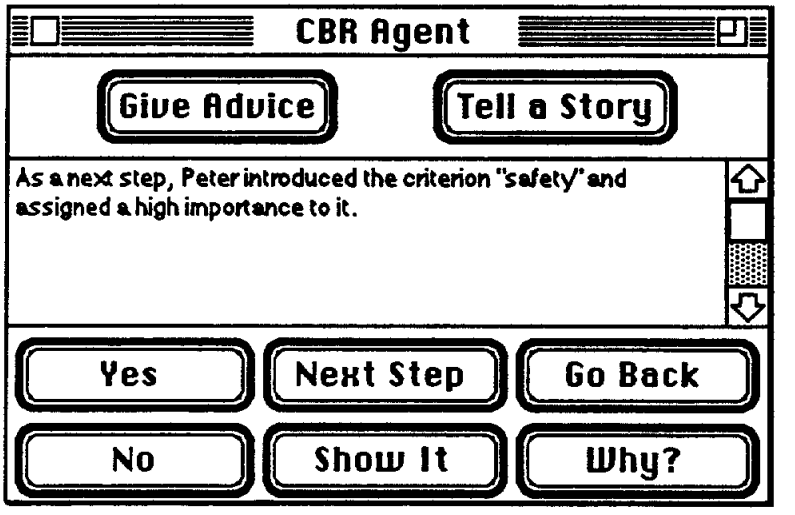

Figure 9: The CBR window.

The MCDSS interface is used by the CBR agent to "tell stories". When requested to do so, the agent freezes and stores the current state of the system and replays a case on the screen. At each step, i.e. at each state transition $(U, P, P_C)_i \Rightarrow (U, P, P_C)_{i+1}$, the user gets back the control of the MCDSS and is able to manipulate the decision situation displayed by the CBR agent. Through the "Next Step" button, the decision maker can require the agent to first comment the event underlying the next state transition (e.g. "As a next step, *U* introduced the criterion *C* and assigned a *W* importance to it" with *U*=Peter, *C*="safety" and *W*=high). The new state is then displayed on the screen (with the related consequence for the ranking of the alternatives) after selection of the "Show It" button. At any time, the user can interrupt the "commented slide show" of the CBR agent and return to the initial state.

In terms of intervention frequency, we distinguish between passive and active interventions:

- **Passive**: the CBR agent intervenes in the decision making process following a user's request ("Give Advice" or "Tell a Story").
- **Active**: the CBR agent can autonomously decide to intervene in the process by assuming one of its two roles (an adviser or a story teller).

In order to avoid frequent interruptions, active intervention by the CBR agent is regulated by a set of rules. These rules determine specific intervention conditions (*triggering configuration* such as "Number of criteria/alternatives too low/too high") as well as thresholds (*frequency thresholds* such as a parameter controlling the time elapsed since the last intervention, and *similarity/relevancy thresholds* to be reached to justify an active intervention). Figure 10 summarizes and provides an overview of the interaction between the decision maker and the CBR agent.

6 Conclusion

The research presented in this paper has outlined how CBR can be used to help build a symbiotic DSS which provides a flexible learning environment for the user. An important characteristic of our approach consists of enabling the system to continuously learn and update itself from user problem solving experiences. We feel that such a two way flow of information is essential for the design of intelligent DSS providing more effective decision support to users.

We are in the process of implementing these ideas in an extension of the *Triple C* MCDSS [2] and we plan to conduct some empirical tests of the impact of CBR on problem solving behaviors of users. We feel that the results of this research will be of benefit to researchers from both the DSS/MCDSS and the CBR communities. This research is the first attempt to extend conventional MCDM techniques in a new type of a symbiotic MCDSS. With respect to the CBR community, it is hoped that the choice of a practical and relatively limited problem (multi-criteria decision making) will yield insights into CBR which will prove useful for more complex and unstructured tasks which are currently tackled in the CBR research literature - with notably limited success to date.

There are many interesting and complex issues raised by the integration of CBR in a MCDSS. The problem described in this paper can be considered as a first step in this integration. Several extensions to this research are possible and are being planned. These extensions include the empirical testing of the impact of CBR on the process of solving MCDM problems and the integration of a causal explanation module in the system.

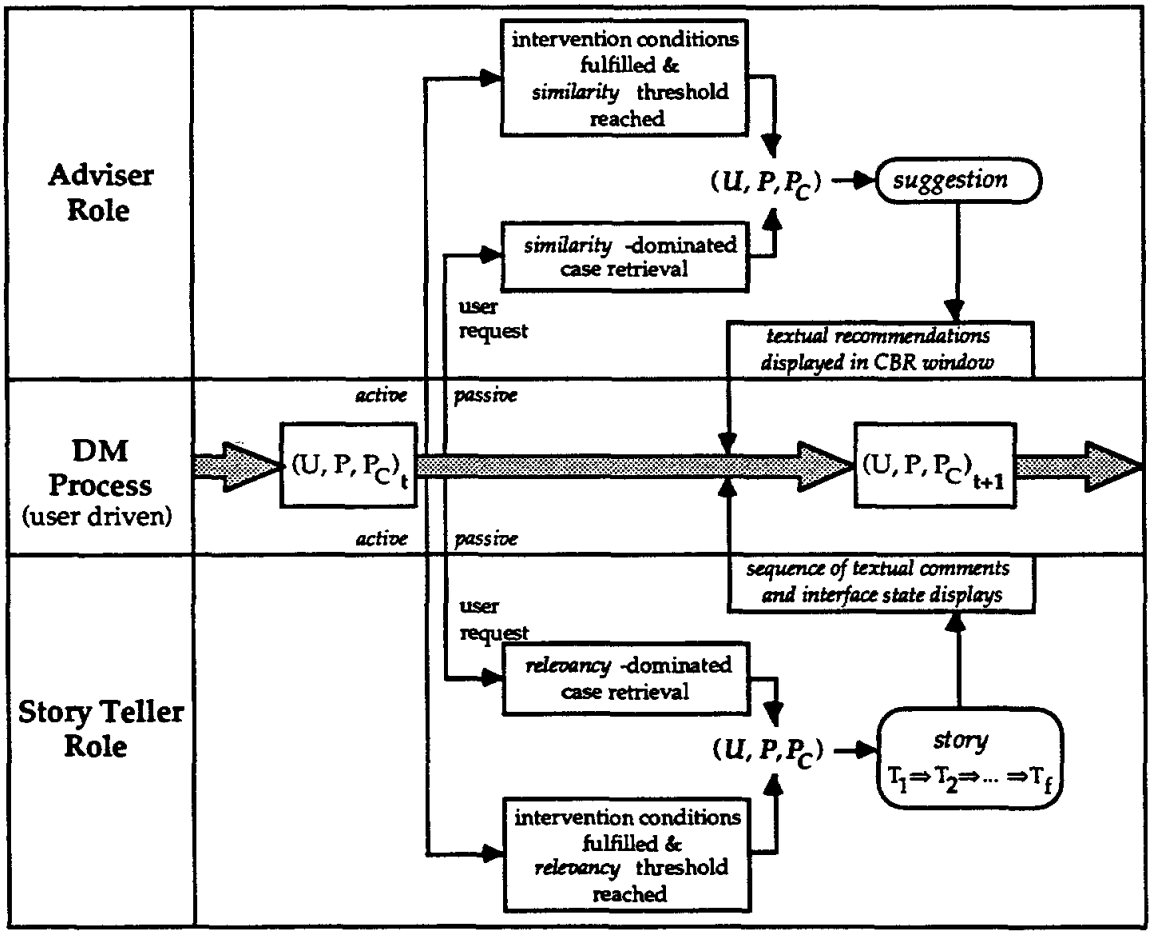

Figure 10: Overview of user-agent interaction.

References

[1] Angehrn, A. A. and H.-J. Lüthi, "Intelligent Decision Support Systems: A Visual Interactive Approach," *Interfaces*, 20, 6 (1990), 17-28.

[2] Angehrn, A. A., "Designing Humanized Systems for Multiple Criteria Decision Making," *Human Systems Management*, 10, (1991), 221-231.

[3] Angehrn, A. A., "Modeling by Example: A Link between Users, Models and Methods in DSS," *European Journal of Operational Research*, 55 (1991), 293-305.

[4] Angehrn, A. A., "Stimulus Agents: An alternative Framework for Computer-aided Decision Making," *DSS-92*, M.S. Silver (ed.), The Institute of Management Science, June 1992.

[5] Bonczec, R.H., C.W. Holsapple and A.B. Whinston, *Foundations of Decision Support Systems*, Academic Press, New York, 1981.

[6] Carbonell, J.G., "Derivational Analogy: A Theory of Reconstructive Problem Solving and Expertise Acquisition," in *Machine Learning, An Artificial Intelligence Approach, Volume II*, Morgan Kaufman, 1986.

[7] Dutta S. & P. P. Bonissone, "Integrating Case-based and Rule-based reasoning: The Possibilistic Connection" in the *Proceedings of the 6th Conference on Uncertainty in Artificial Intelligence*, MIT, Boston, 1990.

[8] Dutta S., "Analogy: An Approximate Reasoning Technique for Answering Null Queries" *International Journal of Approximate Reasoning*,, Vol. 5, No. 4, July 1991, North-Holland.

[9] Eom, H. B., "The Current State of Multiple Criteria Decision Support Systems," *Human Systems Management*, 8 (1989), 113-119.

[10] Fischer, G. and T. Mastaglio, "Computer-based Critics," *Proc. of the 22nd Annual Hawaii International Conference on System Sciences* , 1989, 427-436.

[11] Illich, I., *Tools for Conviviality*, Perennial Library, Harper & Row, New York, 1973.

[12] Keen, P. G. W., "Decision Support Systems: The Next Decade," *Decision Support Systems, The Int. Journal*, 3, 3 (1987), 253-265.

[13] Keeney, R. L. and H. Raiffa, *Decisions with Multiple Objectives*, John Wiley, New York, 1976.

[14] Levine P., J. Pomerol and R. Saneh, "Rules integrate Data in a Multicriteria Decision Support System," *IEEE Trans. on System, Man and Cybernetics*, 20, 3 (1990), 678-686.

[15] Li H.L., "Solving Discrete Multicriteria Decision Problems based on logic-based Decision Support Systems," *Decision Support Systems, The Int. Journal*, 3 (1987), 101-119.

[16] Manheim M.L., "An Architecture for Active DSS," *Proc. of the 21sh Annual Hawaii International Conference on System Sciences*, IEEE Computer Society Press, 3, Hawaii, 1988, pp. 356-365..

[17] Norman, D. and S. Draper, *User Centered System Design: New Perspectives on Human-Computer Interaction*, LEA Publishers, 1986.

[18] Raghavan, S.A. and D.R. Chang, "Exploring Active Decision Support: The Janus Project," *Proc. of the 22nd Annual Hawaii International Conference on System Sciences*, 1989, 33-45.

[19] Riesbeck C.K. & R.C. Shank, *Inside Case-Based Reasoning*, Lawrence Erlbaum Associates Inc., NJ, 1989.

[20] Rissland E. L. & D.B. Skalak, Combining case-based and rule-based reasoning: A heuristic approach, in *Proceedings of 11th Joint Conference on Artificial Intelligence*, San Mateo, CA, Morgan Kaufmann Pub., Inc., Aug. 1989.

[21] Roy B., "The Outranking Approach and the foundations of Electre Methods", in Bana e Costa (Ed.), Readings in Multiple Criteria Decision Aid, Springer Verlag, Berlin, Heidelberg, 1990, p. 155-183.

[22] Saaty, T. L., "Axiomatic Foundation of the Analytic Hierarchy Process," *Management Science*, 32, 7 (1986), 841-855.
[23] Sprague, R.H. Jr. and E.D. Carlson, *Building Effective Decision Support Systems*, Prentice Hall, 1982.
[24] Te'eni, D. and M.J. Ginzberg, "Human-computer Decision Systems: The Multiple Roles of DSS," *European Journal of Operational Research*, 50, 1991, 127-139.
[25] Vincke, Ph., "Multi-attribute Utility theory as a basic approach", in Faudel G. et. al. (Eds.), Multiple Criteria Decision Methods and Applications, Springer Verlag, Berlin, pp. 27-40.
[26] Wilkins D., *Practical Planning*, Morgan Kaufmann Publishers, San Mateo, CA, 1988.
[27] Zeleny, M., "Cognitive Equilibrium: A New Paradigm of Decision Making?," *Human Systems Management*, 8 (1989), 185-188.
[28] Zeleny, M., *Multiple Criteria Decision Making*, McGraw-Hill, New York, NY, 1982.

Appendix 1: The visual interactive interface of the *Triple C* system.

Figure A1 shows a typical *Triple C* screen. The windows displayed on the screen represent different information sources visualized by the user while describing and analyzing a specific decision. In this particular case, the decision-making process is focussed on the choice between different European locations for building and running a new electric power plant.

In a top left window, the system displays a visual representation of the **criteria** introduced by the user. In *Triple C*, each criterion is represented visually by a sector. The size of the sectors indicates the relative importance (weights) assigned to the criteria and can be changed interactively. For instance, in the model displayed, the user seems to give much more importance to cost-related criteria than to factors like safety level, ecology or manpower.

The top right window shows the list of **alternatives** which are considered and analyzed by the decision-maker. The spreadsheet-like representation used in this window allows the user to have a global view of the data relevant to the decision at hand and to easily edit and modify information.

The bottom right window illustrates how the tools integrated in *Triple C* contribute towards giving a **visual dimension** to the different sources of information involved in the description and the exploration of a choice problem. In this particular case, the "Direct Comparison"-tool displayed on the screen supports the user in comparing alternatives visually and in better understanding the similarities and the differences between them (for instance between the two locations France and Portugal).

In the bottom left window, the system suggests a possible **ranking** of the alternatives. This ranking is interactively calculated by the system using a mathematical technique and is updated every time the decision-maker either changes the weights of the criteria or introduces new ones. As a special characteristic, *Triple C* also allows decision-makers to modify the ranking of the alternatives according to their individual judgement. Every time the user changes the ranking (by replacing the alternatives in the "Ranking"-window) the system automatically computes - via an optimization algorithm - a new set of weights for the criteria. In this way, *Triple C* also supports decision makers in finding **arguments** or

justifications for a specific choice, i.e. in answering the question: "Given a certain ranking of the alternatives - e.g. France should be the best alternative - how should the different criteria be weighted?" By answering this kind of question and by performing sensitivity analysis (changing the weights of the criteria and analyzing the consequences), users can gain more insights into the decision situation at hand and incrementally learn about their own subjective view and **preference** structure.

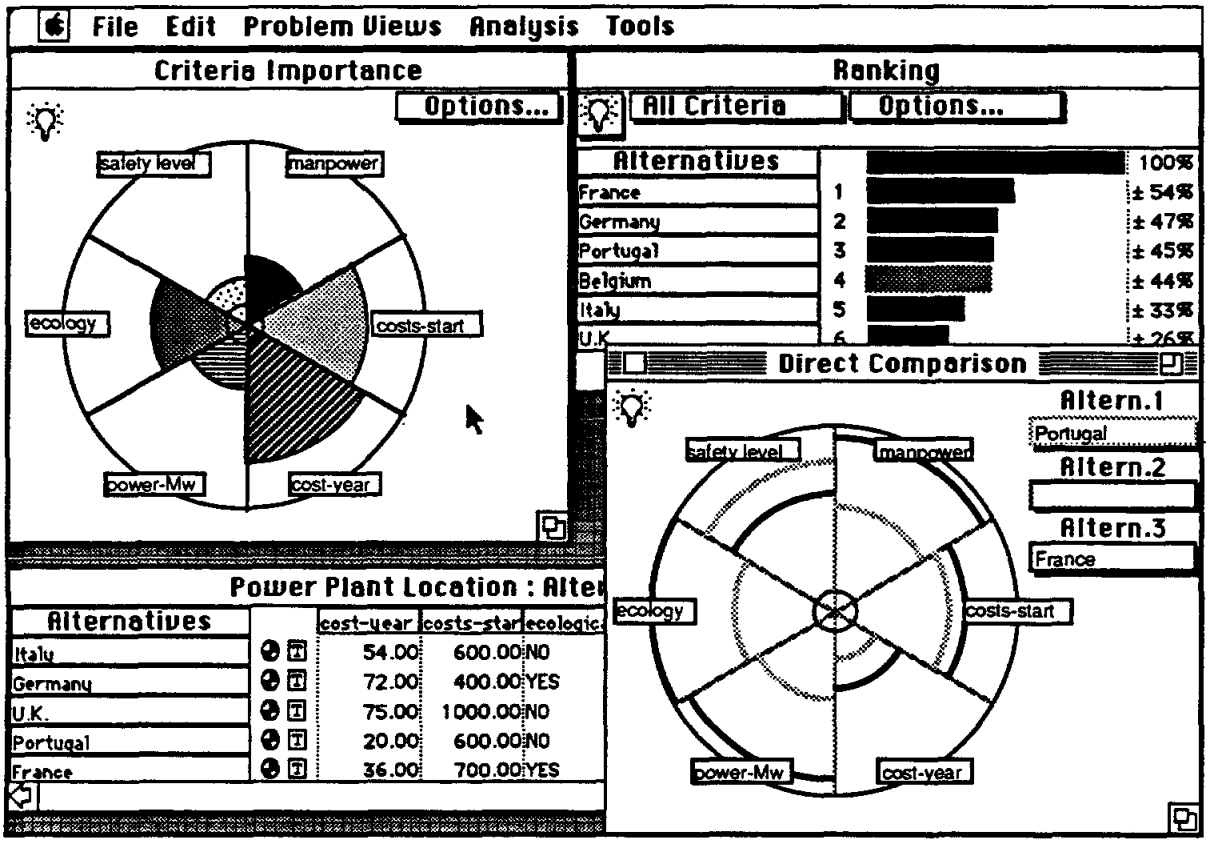

Figure A1: A typical screen of the *Triple C* system

Appendix 2: The *Problem Browser Tool*

Figure A2 shows three screens of the **Problem Browser Tool (PBT)** described in section 5. The two screens on the left hand side of Figure A2 illustrate how the PBT can be employed as a **hypertext navigation tool** to support browsing in the Case Library (CL). The PBT also reflects the hierarchy of problems described in section 4: By selecting one of the existing problem domains (e.g. "Software Selection") the user accesses a selected list of cases stored in the CL. As a next step, the decision maker can either access information about a specific case (e.g. a case of DSS software selection) or activate the story teller component of the MCDSS. For this purpose, the PBT is dynamically linked to the *Triple C* system discussed in Appendix 1.

Figure A2 also shows how the CBR agent encapsulated in *Triple C* employs the PBT to collect information needed to create and update the relevancy matrices described in section 4. For instance, as displayed on the right hand side of Figure A2, the decision maker's objective could be to explore a new software selection problem involving the choice of a CAD system. In this particular case, the PBT serves to flexibly and automatically elicit information about the relevancy of existing cases (Word Processor's choice, Spreadsheet's choice, etc.) for the new decision situation addressed by the MCDSS user.

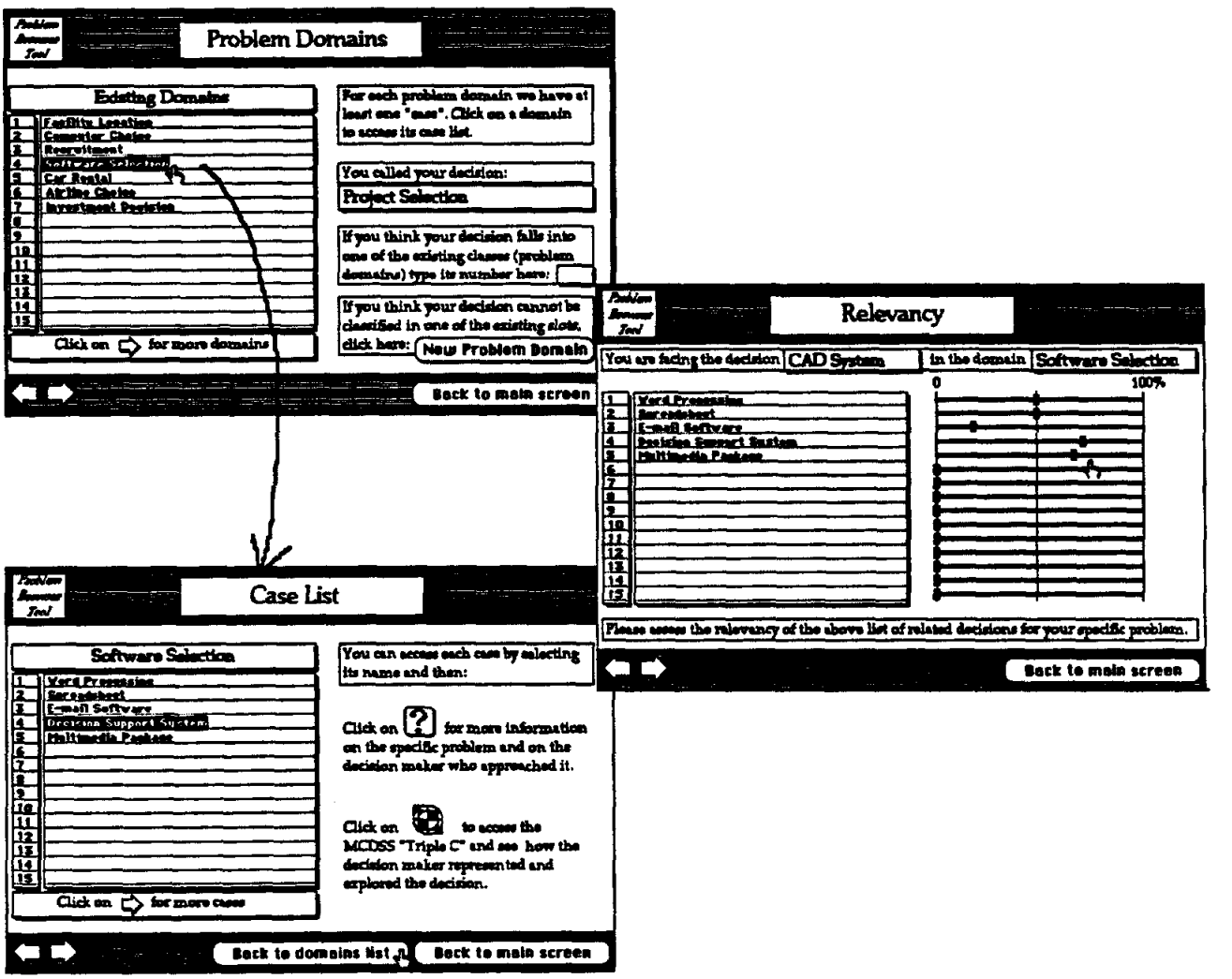

Figure A2: Screens of the Problem Browser Tool (PBT)