Abstract

This paper describes our research approach for developing active decision support systems. We provide a detailed discussion of our research framework, methodology, and system architecture for developing advanced forms of decision support and intellectual support. We present the details of the JANUS system, our research prototype, where we have implemented our ideas. We discuss the contributions of the JANUS work and summarize our experiences and plans for the future.

1. Purpose and Organization

This paper has two objectives: Presenting our research ideas and approaches for developing active decision support systems; and describing our research prototype called JANUS that implements our ideas.

The rest of the paper is organized into five sections. Section 2 provides an introduction to the concept of active decision support and reviews the key works from the literature. Section 3 describes our research framework for exploring active decision support ideas. Section 4 is devoted to the detail description of the JANUS system. The system is described in terms of the goals and objectives, functional features, design overview, and a sample session. In Section 5, we establish the unique contributions of the JANUS project by establishing its relationships with relevant prior work. We also reflect on the outcomes of the JANUS project, provide a list of the key research problems that have emerged, and discuss our plans for future work. The final section summarizes the key aspects of the paper.

2. Active Decision Support

2.1 Introduction

The concept of active decision support systems represents a variation and refinement of the fundamental DSS philosophy [Gory71]. Whereas the DSS philosophy merely calls for enhancing human decision making through computer-based tools, the active decision support concept advocates developing advanced forms of decision support where the underlying tools actively participate in the decision making process.

The notion of active participation in decision making can represent a broad range of ideas such as monitoring the decision making processes of the user and detecting inconsistencies and problems; automatically scheduling and carrying out the necessary activities by understanding the intent and the context of the decision maker; alerting the decision maker to the aspects of the problem and problem-solving processes he may be unconsciously ignoring; carrying on conversations with the decision maker that can lead to proper formulation of decision problems; stimulating creative ideas; serving as a sounding board for ideas; and criticizing decision maker's actions and decisions from various perspectives.

Active decision support ideas stand in striking contrast to the approaches underlying the conventional DSSs. The latter are largely passive partners in decision making. They are passive in the sense that they merely place a set of useful facilities at the disposal of a decision maker. They expect that the decision maker will somehow exploit these facilities effectively during decision making. They are not capable of taking initiatives - they can only respond to users' requests. In summary, they provide a weak form of support that does not exploit the full potential of a computer-based system.

2.2 Underlying premises

The key premises underlying the active decision support research can be summarized as follows:

- Decision support systems are essentially man-machine systems for improving decision making [Raghavan87a]. Since man and machine have distinguishing characteristics and skills, a potentially promising synergy exists between them. This synergy can be realized by properly distributing the roles, skills, and responsibilities between the man and the machine within the man-machine setup [Woods86].

- It is possible for the machine component to play active roles during decision making without violating the fundamental support philosophy of decision support systems.

- It is necessary to incorporate active roles in the machine component for fully exploiting the potential power of a computer.

- AI/Expert systems technology can provide the necessary implementation techniques for implementing active support ideas.
2.3 A Review of the Key Ideas

Research in active DSSs is carried out under a variety of labels such as intelligent decision support systems\[Hollangel86\], symbiotic DSS[Manaheim88], and joint man-machine cognition\[Woods86, Raghavan87a\]. Currently there are four broad threads of ideas in this area: idea stimulation, autonomous processes, expert systems, and active elicitation and structuring.

2.3.1 Idea Stimulation

Idea stimulation is widely recognized as a very important form of providing active decision support\[Young82, Manaheim88, Krcmar86, Neirenberg87, Raghavan87\]. There are at least two systems that illustrate this approach\[Krcmar86, Neirenberg87\]. Later we will also discuss how this approach is pursued in our JANUS system.

Krcmar\[Krcmar86\] have developed a DSS that can help users identify new ways to exploit information technology as a competitive weapon. They use questions as triggers for stimulating new ideas. They generate the trigger questions by using a theoretical model that is widely used for studying information technology and its impacts.

The underlying model provides primitive variables for characterizing information technology, impacts, and their inter-relationships. Each relationship in this model represents a potentially new idea for exploiting information technology as a competitive weapon. This provides a basis for stimulating new ideas - facilitating the user to think about the potential relationships between the variables in the model. The system accomplishes this by systematically instantiating the model variables, and posing questions about the possible relationships. Since the number of questions at any point in time can be combinatorially explosive, the system uses contextual information for pruning down the irrelevant ones. The authors do not provide any system performance measures.

Whereas Krcmar uses a problem-specific model for idea stimulation, \[Neirenberg87\] employs a set of domain independent modules for stimulating ideas. Their system, named Idea Generator, is essentially a decision structuring tool. The underlying technique uses simple primitives such as problem, goal, actions, and strengths of relationships for structuring a decision problem. The system uses idea generation modules for helping the user identify novel actions.

Each module in the system is based on a specific scheme for generating novel actions. Some of the schemes used by the modules are:

- Think of similar situations.
- Think of metaphors for the situation.
- Think from other perspectives - that is think of how other people may solve the problem.
- Focus on goals one at a time and then collectively.
- Reverse your goals and actions.
- Focus on the people who will be affected by your actions.

The user can collect the ideas that are generated into a temporary workspace. The system provides facilities for grouping, pruning, and synthesizing these ideas. Authors claim that the system has been used in several simple business problems and has proved to be quite effective.

2.3.2 Autonomous Processes

Here the active support is implemented as a set of daemons or agents that watch over the decision making process of the user and trigger appropriate responses autonomously. In \[Raghavan84\] we proposed several initial ideas in this direction: observing decision maker's activities and scheduling the necessary related tasks; keeping track of the pending tasks and ensuring that they are completed; eliciting and enforcing constraints; forcing a divergent process if the user is judged to be prematurely converging; and forcing a convergent process if user is deemed to be getting disorganized with too many tasks and thoughts.

Recently, Manaheim\[Manaheim88\] has proposed a general architecture for active DSSs based on autonomous processes. The key aspect of his architecture is the existence of two kinds of processes in the system: user directed, and system directed. User directed processes correspond to tasks in conventional passive DSS. The system directed processes, on the other hand, are processes that are autonomously initiated by the system while playing its role as an independent and active agent in the decision making process.

The ability of the system to play active roles in this architecture rests on the following critical factors: having a good understanding of the decision making processes of the user; having a normative criteria for judging the decision making process; and having strategies for improving the process. Once these requirements are met, the system can closely monitor the decision making process of the user and intervene when necessary to criticize and offer suggestions. It can raise pointed questions and force the user to think of other alternatives and to provide rationale and justifications for his actions. It can also anticipate users needs and schedule processes in advance.

Since Manaheim's architecture rests on having an explicit model of decision making process (he uses the term problem-working process), much of his work \[Manaheim87\] focusses on developing the necessary theoretical bases. At present there are no prototype systems to demonstrate his ideas.

2.3.3 Expert Systems as Active DSSs

One could argue that every expert system is an active DSS because it can be used merely for getting advice
Miller84 provides a comprehensive description of the ATTENDING system, a critiquing expert system for the medical domain. Here the system becomes active only after the user has a tentative decision. The system interacts with the user and gathers the details of the problem, his decision, rationale and justifications. This dialog process itself can be very insightful to the decision maker, as he is forced to communicate and justify his decision to the system. After the details are collected, the system reconstructs a plausible decision making process using its knowledge base and internal models, and identifies potential problems and possible improvements.

A closely related approach is to capture the generic reasoning processes of different problem-solving perspectives for the purposes of critiquing. For example, a decision maker can greatly benefit by getting help from a structuring technique. Since structuring techniques are valid active support ideas, they are less interesting from our perspective and therefore not discussed further.

2.3.4. Active Problem Elicitation and Structuring

Here the system is based on a structuring technique that is suitable for the problems of interest. For example, goal-oriented structuring, analytical hierarchy structuring etc. Since structuring techniques are normative models of decision making, they immediately provide: a basis for active problem elicitation, a basis for making recommendations, criteria for judging the decision making process, and a framework for incorporating idea stimulation and other machine-based personalities. Thus this approach makes it easier to implement many of Manheim's ideas by avoiding the problem of understanding the unconstrained decision making process of the user, a critical requirement of his architecture.

The key role of the system that is based on this approach is helping the user organize and structure their own knowledge and expertise effectively for solving problems. The GODESS system[Pearl84] is an excellent example of such a system. The acronym GODESS stands for goal-oriented decision structuring system. Goal-oriented structuring is an adaptation of means-ends analysis, a technique that is widely used in AI planning systems. Here a problem is structured in terms of goals, actions, preconditions, states, factors, and strengths of relationship between these components.

GODESS can play both support and decision-making roles. In the support role, the system carries on an active dialog with the user and formulates the decision problem in terms of the primitives of the goal-oriented structuring technique. Since the system is domain-independent, its only knowledge is that of the structuring technique. It relies on the decision maker to be very knowledgeable about the problem and supply the problem-specific knowledge.

GODESS structures the details of the problem, as they unfold, around an And-Or tree. This tree is used throughout the dialog process for meaningfully communicating with the user. Making decisions about how the focus should shift between various parts of the problem, and determining what aspects of the problem need further elaboration. At the end of problem information gathering, the system processes the information accumulated in the And-Or tree to make recommendations.

The GODESS work adds several key ideas for developing active decision support: active problem elicitation and decision structuring; domain independent decision support; exploiting users' knowledge of the decision problem; and adapting AI problem-solving techniques for decision structuring. Though the system does not use any specialized domain knowledge, its architectural framework does not preclude the incorporation of knowledge bases.

2.4. Summary

We discussed four broad themes of ideas for developing active decision support: idea stimulation, autonomous processes, expert critiquing systems, and active elicitation and structuring techniques. Though we described them as disjoint ideas, they are closely related to each other and can be easily combined together.

3. Our Research Framework

3.1 Overall Context and Goals

The overall context of our research is developing decision support environments where man and machine can engage in an effective partnership during decision making. We visualize these environments as integrated
set of tools and facilities that operationalize the various alternative strategies for decision support.

We recognize three major strategies for developing decision support: resource support, process support, and intellectual support. In the resource support approach, the focus is on providing the resources that are necessary for decision making. In the process support approach, the focus is on addressing the generic needs of decision making processes. In the intellectual support approach, the thrust is to enhance the creativity of the decision maker. The operational level goals of these approaches are summarized in Tables 1, 2, and 3.

In our research, we concentrate on process support and intellectual support approaches. Our goal is to resolve the conceptual and implementation problems underlying these approaches. We do not address active support as an explicit goal, as it is a recurring theme throughout our research.

3.2 Methodology

The key intellectual problem in our research is bridging the gap between the conceptual ideas and implementation techniques. There are uncertainties associated with both the ends. We cannot be certain that our ideas represent valid abstractions, are at the right level, and carry enough content and direction for developing implementations. We are also not certain that our ideas can be implemented using the available implementation techniques. This is akin to situations that normally occur in the artificial intelligence (AI) research. Therefore, we use the exploratory systems development paradigm of research that is generally used in AI for tackling such problems. The paradigm is founded on the following premise - implementing ideas on a machine makes exacting demands on rigorosity, conceptual clarity, and resolution of details; therefore getting a working implementation is a very valuable research process and a good test of the ideas.

In this paradigm, the research process proceeds as follows. The researcher begins by developing an exploratory prototype. He then uses this prototype as the experimental vehicle/environment for refining and overcoming the problems associated with his ideas and implementation techniques. He evolves the prototype continuously as he recognizes and resolves the problems. When the research is completed, the prototype represents a clear expression of both the problems and solutions to the problems. The prototype plays a critical role in the research process as well as becomes a significant output of the research.

Within this overall paradigm, we use the following step-by-step approach:

Pick a good structuring technique. Use it as the basis for the steps that follow.

Table 1: Useful Resources for Decision Making

- Supporting the planning, organizing, and the execution of complex and inter-related tasks that constitute decision-making
- Supporting flexible process sequences during decision making
- Supporting interruption and resumption
- Simulating decisions and studying their potential consequences
- Supporting multiple worlds/contexts for exploring potential scenarios
- Providing various schemes for choice reduction
- Maintaining the details about intermediate decisions and their inter-relationships

Table 2. Aspects of Process Support Approach

- Active elicitation and structuring of problems
- Surfacing the assumptions, justifications and contingencies
- Stimulating creative ideas, learning, and discovery
- Suggesting alternatives and improvements
- Critiquing decision makers' processes, judgments, and decisions
- Overcoming decision makers' tunnel vision, fixations, and biases
- Promoting convergent and divergent thinking
- Employing machine-based personalities for analyzing problems from diverse perspectives
- The machine playing various kinds of sounding board roles. For example: playing a devil's advocate

Table 3. Aspects of Intellectual Support Approach

- Data
  - Models
    - Statistical models
    - OR/MS, optimization models
    - Other Quantitative models
    - Qualitative and Symbolic models
    - Causal models
    - Knowledge bases
    - Domain specific
    - General heuristics
    - Machine-based experts
    - Expert consultants
Develop a machine representation for organizing the problem details.

Enhance the representation for primitives such as assumptions, justifications, and contingencies.

Develop elicitation strategies.

Implement the process support mechanisms.

Identify criteria and strategies for improving the decision making process.

Incorporate machine personalities that can stimulate thinking and enhance the problem solving process.

Develop knowledge bases needed for the machine personalities. Develop modules for what-if, sensitivity, and contingency analyses.

3.3 Conceptual Architecture

The conceptual architecture we use for our exploration is shown in Figure 1. We recognize four major functional components: Representation, Elicitation, Analysis, and User interface. Representation provides the schemes for representing information internally, and mechanisms for retrieval and inferencing. It also provides the base for organizing and utilizing domain-specific and generalized knowledge bases. Analysis block provides for internal consistency checks, constraint satisfaction, triggers and daemons, sensitivity analysis, scenario analysis, and decision simulation.

Elicitation carries dialog generation strategies, and active agents for critiquing, stimulating ideas, and playing partnerships roles. User interface provides high bandwidth communication, process support mechanisms for: mixed-mode initiative i.e. the ability to shift control between the system and the user freely, flexible process sequences, interruption and resumptions.

The structuring methodology underlying the system is the conceptual glue that unifies the architectural components of the system. It provides the primitives and vocabulary for thinking about and articulating domain problems, graphical primitives for man-machine communication, rules for internal consistency, methods for analysis, and bases for making recommendations.

4. The Description of the JANUS System

4.1 An Overview of the Functional Features

JANUS is an experimental research prototype we have developed for exploring our research ideas. The system (~8000 lines) is implemented in C-Prolog under VAX/Ultrix. The system uses Saaty's analytical hierarchical process as the underlying decision structuring methodology.

The primitives of the analytical hierarchical process are Goals, Factors, Subfactors, and Judgments. Figure 2 illustrates the relationships between these primitives through a simple decision problem. In the JANUS system we have added three additional primitives—Notes, Justifications and Contingencies. Note is an arbitrary piece of text or annotation that can be attached to any element of the representation. The justification primitive is used for capturing the assumptions and premises underlying the elements of a decision problem. The contingency primitive is used for capturing the relationships between the elements of the problem and the exogenous variables.

The following is the summary of the major functional features/capabilities implemented by the system:
Elicitation

- Active elicitation and structuring of the details of a decision problem i.e. goals, factors, alternatives, and judgments.

- In addition to the essential details, the system is also capable of eliciting and capturing the assumptions, justifications and contingencies behind users statements.

- A keyword facility automatically extracts key words from the input text, and organizes the text for flexible retrieval.

- The system provides a notepad facility for creating and retrieving notes. The notes are arbitrary pieces of text that can be attached to any element of the problem. They can be used in a variety of ways. For example, leaving reminders to oneself; jotting down important ideas for later use etc.

- A constraint/contingency entry and checking facility for imposing constraints and contingencies on judgments made by the decision maker.

Process Support

- The system supports dual initiative. That is, the overall decision making process can be controlled either by the user or by the system. The control can be freely switched between them at any point in time.

- The system supports flexible process sequences. That is, the decision structuring process is not constrained to follow any predetermined order. Though the underlying structuring strategy has a top-down orientation, the system permits a high degree of flexibility regarding the order in which the details are collected. For example, the user can describe factors before alternatives and vice versa. He can also defer answering questions if he feels a need to do so. The system will keep track of the resulting loose ends and raise questions at the appropriate later times.

- The system supports interruptions and resumptions by providing mechanisms for saving and restoring problem contexts. This mechanism is also used for by the system for accumulating its data base of solved problems.

- Users can customize the various aspects of the structuring process through the user programmability feature.

Decision Recommendation & Analysis

- The system can make decision recommendations by performing the calculations defined by the analytical hierarchy process.

- While describing/explaining its recommendations, the system can integrate the relevant notes, assumptions, justifications and contingencies.

- The user can perform what-if analyses by changing the values of the judgments and contingency variables.

Mind Expansion

- The system provides intellectual support through a "mind expansion module". The module supports four different personalities/roles: Spock, Bozo, Mom, and Aesop. Spock emulates logical thinking. Bozo emulates lateral thinking. Mom emulates a personality that is obsessed with justifications and counter arguments. Aesop is a story teller.

Knowledge Base

- The system provides facilities for developing a database of solved problems for use by the mind-expansion module for suggesting new ideas to the user. The problems are organized and indexed using the primitives of the structuring model and key words.

- The knowledge base of the mind-expansion personalities are represented in a declarative form using templates. This facilitates the ease of evolving and enhancing the knowledge bases.

Customization

- A keyword facility for defining keywords and complex relationships among keywords such as syntactical synonyms, semantic synonyms, antonyms, and hierarchies.

- The system provides appropriate entry and exit points for the user to define daemons and insert code segments for customizing the various aspects of the structuring process.

4.2 Design Overview

4.2.1 Architecture

The architecture of the system is shown in Figure 3. At the overall level the modules in the system are: user-interface, scheduler, action-cluster, and representations manager. The user interface module is responsible for low-level system-user interactions. It gathers user inputs and displays messages. The scheduler module controls the overall operation of the system. It consists of an agenda structure, a scheduling logic, and a work data base. The agenda structure is used for keeping track of the pending tasks. Modules in the system invoke each other by placing task requests in the agenda. Any task that
The Scheduler and the Work Data Base can occur in the system has a priority level and a generic task description. The priority levels are assigned (at the design time) to reflect the natural order in which the problem elicitation should proceed. The scheduling logic determines how the tasks are selected for execution. The default selection criteria uses only priority levels. However, the scheduler is capable of using any other criteria specified through the customizing code.

The scheduler also carries daemons (there are none at present) which watch over the representation, the agenda, and the works data base. When a daemon fires, it becomes a task in the agenda and gets selected as per the normal scheme of things.

The work data base within the scheduler is used for various purposes: storing secondary details of tasks (only primary details are stored on the agenda), maintaining the overall system context information, and as a message exchange area by the modules of the system.

The action-cluster carries all the worker-modules of the system. As a general rule, there is one worker module for each major activity in the system. A partial list of the worker modules are: goal-entry, factor-entry, alternative-entry, judgement-entry, mind_expander, problem_analysis, constraints-entry, and notepad-entry. The scheduling modules invokes the appropriate worker module when a task needs to be completed. Usually worker modules schedule tasks for other modules when they complete. For example, the last step in the goal_entry processing is scheduling tasks for factor_entry, and alternatives entry.

The representation manager is responsible for providing a higher level interface that shields the low level implementation details of the representation from other modules. It maintains the decision problem information, keywords data base, and mind expansion knowledge bases. The keywords data base maintains the keyword taxonomies, syntactical synonyms and semantical synonyms, and links to the textual information contained in the problem structure and knowledge bases. The mind expansion knowledge base maintains a library of reference problems, text templates, stories for the aesop module, and specialized rules for matching and retrieval.

4.2.2 System Operation

The basic execution cycle of the system is as follows. The scheduler picks up a task from the agenda and invokes the appropriate worker module. The worker modules gets the details of the task from agenda and the work data base. Typically it accesses the representation to retrieve the relevant details of the problem. It may then schedule a task for the user interface to collect some input from the user. As a last step in its processing, it may reschedule itself for processing the user input.

The user interface module is then invoked. It interacts with the user and places the users input in the work data base and relinquishes control back to the scheduler. Now the earlier worker module gets reinvoked for processing of the user input. This time the module may schedule appropriate tasks for other modules before completing.

This basic cycle repeats. During this cycle, the user may force his initiative any time the user interface is waiting for his input. By issuing a command he can request any task of his choice to be performed. The command translates into a high priority task on the agenda and gets scheduled next. When the task all its related tasks are completed, the control returns back to the system automatically.

If the user requests mind expansion functions, then the underlying personalities are invoked in sequence. The personalities engage in an appropriate conversation to stimulate the user. They use the overall context information from the scheduler, the problem details that has been gathered till that point in time, and their own internal context to guide their conversation. For example, if the user is working on factors for a personal computer purchasing problem, the mind expander may show him the factors that were associated with an equipment selection decision that was made in the past.

Whenever the user inputs text say for a goal, or factor, first the representation is updated. Then the keywords in the text are automatically extracted and retrieval links are established for promoting flexible retrieval. The system provides various alternative ways to extract a piece of text: using the conceptual primitives like goal, factor etc.; by traversing the problem graph, for example, getting to the factors from the goal; by using the keyword mechanism—either by direct match or by matching through synonyms, antonyms, closeness measures etc. These
flexible retrieval mechanisms provides the base for the operation of the mind expansion personalities.

4.2.3 Knowledge Representation

The basic structures in the knowledge base are: trees for storing problem information and key word taxonomies; lists for pointers from keywords to the various conceptual entities; templates and rules for text generation; and production rules for specifying situation-action pairs. The standard prolog primitives of relations, lists and rules are used for implementing these structures.

Since the overall control structure of the system, i.e. the scheduling operations, are data-driven, we have implemented a simple forward-chaining interpreter on top of the backward-chaining scheme of the prolog engine.

4.2.4 Mind Expansion Module

The system supports four distinct mind-expansion personalities: Spock, Bozo, Mom, and Aesop. Each personality pursues a characteristic strategy for helping the decision maker.

Spock emulates logical thinking. It uses the keywords associated with the current problem context to retrieve similar problems from its data base to trigger ideas. Its slogan is "If problem A is similar to problem B then the knowledge of goals, factors, and alternatives of the problem A can be useful for solving problem B".

For example, if the user is working on the problem "Buying a Boat", Spock may suggest that Cost, Loan Rate, Resale value as relevant factors by extracting them from the problem of "Buying a Car".

Bozo emulates lateral thinking. It forces the user to develop alternative problem formulations by tracing the goals-means hierarchy, by thinking about incorrect and infeasible formulations and how they may be made feasible. Its uses several slogans: "A goal is a means to a higher level goal", "Any problem A is contained in some other problem B; therefore solving problem B solves problem A", "A good way to come up with novel alternatives is to think of non-alternatives and think clearly why they do not work and how they can be made to work", and "Think of at least N alternatives before making a choice", "Think of what alternatives some person X would come up with".

Thus Bozo will raise questions such as: Well are there alternatives which you would definitely not consider? Why? Under what conditions will they work? Under what conditions will this alternative not work? If Car is to Transportation, What is X's problem? is to ?

Mom emulates a personality that is obsessed with justifications and counter arguments. Its slogans are: "My son has not thought about all possible cases. I have to raise counter examples", "He is making assumptions", and "His justifications could be wrong". So it poses potential alternatives and makes the user think about why they should be rejected or accepted. It also forces the user to justify his factors, alternatives and judgments.

Thus in the context of "Buying a Computer" problem, Mom may use the keywords taxonomy to find out that words such as calculator, Macintosh, VAX are related to computer, and raise questions such as "How about Macintosh?", "Why do you think service is important than price?" etc.

Aesop is a story teller. It presents the user with short parables that have general morals for problem-solving. The intent is to help the user overcome his fixations and tunnel visions. For example, it may present the story of Newton making the egg stand on its ends by placing it on a ring to warn the user that he may be unduly constraining himself.

The user can invoke these personalities any time during elicitation. The mind expansion module attempts to use the available contextual information to generate appropriate idea stimulating responses. The system remembers the phrases it has used before and avoids repeating them during a given problem.

4.3 Sample Session

The following is an edited excerpt from a sample session with the system. To keep the illustration short, we focus mostly on the interaction with between the user and the mind expansion module.

Legend: > - Janus system messages
      / - our annotations
      rest are texts typed by the user.

> Welcome to JANUS system
> What is your goal? (return = abort, help=help)
I want to buy a computer.
> Give factors for: "I want to buy a computer"
> Cost
> Adding factors to: "I want to buy a computer".
> 1. Cost
> Availability of programs.
> Adding factors to: "I want to buy a computer".
> 1. Cost
> 2. Availability of programs

1 [User is invoking idea stimulation]
I have problem information for the following:

It may aid you in identifying more factors:

1. I want to buy a boat.
2. I want to buy a house.

It seems logical that one of them may help you find a factor. Indicate the item of interest.

For the goal: "I want to buy a house."

I have the following factors:
1. Cost
2. Location
3. Interest rate

And I have the following alternatives:
1. Condo
2. Farmhouse
3. Split

Would you like to see more? (Yes/No)

No

Adding factors to: "I want to buy a computer."
1. Cost
2. Availability of programs

Reliability of the machine

Adding factors to: "I want to buy a computer."
1. Cost
2. Availability of programs
3. Reliability of the machine

Would you like to add some more factors to this goal later? (Yes/No):
Yes (System will come back to ask for more later)

Enter alternative for "I want to buy a computer"

IBM PC
5. Discussion

5.1 The Project Goals and Outcome

The objectives for this project were: Exploring process support and intellectual support; developing a good understanding of the underlying conceptual and implementation problems; developing strategies for resolving these problems; and developing a research prototype that can be used for on-going research exploration. The project has succeeded well on all these counts.

We have been able to translate much of our process support and intellectual support ideas into working features of the JANUS system. Specifically, we have successfully implemented the following features: active elicitation, capturing contingencies and justifications, the system keeping track of pending tasks, flexible process sequences, mixed-mode initiative, and idea stimulation through machine-based personalities. Though the current implementations of these features are rudimentary in nature, they provide a good starting point for further exploration.

We have not conducted a formal evaluation of the performance and effectiveness of the system. But, our preliminary and informal evaluations are very reassuring. We have verified that the system makes correct decision recommendations as per the analytical hierarchy model. We have received good feedbacks from friendly-users who have used the system for simple decision problems. Their experiences indicate that the features of the system are well-founded and effective.

The system has several limitations that are customary of prototype systems. The user interface is very basic and crude. The problem can be traced to two sources. First, the use of main-frame environment - notorious for their user interfaces. Second, since developing good user interfaces would have diverted our limited resources from other aspects we were exploring, we did not give it a high priority. We are currently remediating the interface problems by porting the system to run under Arity-Prolog on IBM-AT compatibles and to take advantage of the extensive user interface capabilities available in that environment. The knowledge base and keywords data base of the system are at present very small. This greatly limits the performance of the mind-expansion personalities. Our plan was to evolve/expand the underlying knowledge bases slowly over time. But we are recognizing the need to start with a sizeable knowledge base and keywords data base in the system. Therefore we have started a mini-project for populating the knowledge base with: representative decision problems; adages and stories related to problem solving; a library of probing questions for stimulating lateral thinking and problem reformulations; and adding a keywords dictionary.

On the whole, the project has been a rich learning experience in terms of understanding the underlying conceptual and implementation problems. We will talk more about these in Section 5.3. For now, our experiences so far have been very encouraging. We seem to have made the right decisions regarding the architecture and the choice of Prolog as the implementation language. The system has taken shape as an easily-extensible system, and promises to be a good research vehicle for our further exploration.

5.2 Relating to Prior Work

The JANUS work integrates and extends several current approaches to active support. [Krcmar87] and [Neirenberg87] explore the use of questions as mechanisms for stimulating ideas. In [Krcmar87], questions are generated using a model of the competitive analysis problem. The advantage of this approach is the specificity of the questions. However, the system may bias the user to think within the framework of the model. Further, the system becomes problem-dependent. On the other hand, [Neirenberg87] uses generic questions that are independent of the problem domain. While this can promote divergent thinking, it lacks the specificity enjoyed by Krcmar's approach.

In JANUS we have tried to get the best of both these worlds. We use a generic structuring technique to guide problem-solving. Therefore the system can be effective for a wide range of problems that can benefit from the underlying structuring strategy. The questions we raise originate from this model. Therefore they can be focused and efficient. At the same time, since the model is generic, the questions can apply to a variety of situations. Further, our mind expansion facilities try to use the problem context information to generate questions and locating analogous situations. Our architecture supports a knowledge base that can be loaded with information...
specific to the kinds of problems expected to be solved using the system.

Some of the features in JANUS are essentially operationalizations of the conceptual ideas outlined by Manaheim[Manaheim88]. In JANUS, the user is not constrained to execute his tasks in any predetermined chronological order. As the user performs tasks, the system sort of watches over his activities and schedules related tasks which have to be completed. The system periodically reminds the user of tasks that are pending and need to be completed. These are operationalizations of the system-directed processes ideas of Manaheim. The system also has daemons which can enforce constraints and consistencies. At a later date, we hope to use the daemons to change the system mode dynamically to force the user to think divergently or convergently[Raghavan84] as needed.

A critical requirement of Manaheim's architecture is to have a good understanding of the users decision making processes. Given that decision making is an extremely complex process, we believe that it may be extremely difficult, if not impossible, to meet this requirement. We feel that the most productive approach to use a good generic structuring technique, and implement it with appropriate process support mechanisms so that user is free to carry his processes in a flexible and more or less unconstrained manner. We have illustrated this approach through the JANUS system.

Though JANUS is similar to GODESS in the sense that both are essentially generic structuring systems, they are considerably different from each other in terms of the support they provide to a decision maker. JANUS goes far beyond GODESS because it not only provides structuring support like GODESS, but also provides process support mechanisms and mind-expansion personalities. The popular approach for enhancing the performance of a generic structuring system is to incorporate domain-specific knowledge. In JANUS we have shown other alternate approaches for enhancing the support capabilities of a generic structuring system.

At present, our mind expansion personalities are not critiquing agents, or expert personalities in the sense of our discussions in "employing expert systems as active DSS" (Section 2.3.3). But JANUS provides the overall architectural framework for productively exploring these ideas. A project is underway to implement mind-expansion personalities along the lines of PARRY[Colby75] and POLITICS[Carbonnel80].

In summary, the JANUS work shows how the various active support approaches can be integrated and extended. The work is significant from many perspectives: exploration of process support mechanisms and mind expansion facilities; exploration of active support in a generic fashion; and the methodological thrust - the use of bottom-up strategy for developing conceptual and implementation ideas in parallel.

5.3 Future Work

During the course of the project, we have acquired valuable insights into the problems that underline the development of man-machine joint cognition systems. We have developed a long-list of research problems that need to be investigated. These will be the focus of our on-going research. Since the list is rather long, we will discuss only some of the key problems.

The mind-expansion personalities in the current system are based on simple and intuitive models. As a result, their capabilities are quite limited. They can only 'speak'; cannot 'listen'; are not capable of understanding and utilizing the responses of the user, and integrating them in their on-going dialogs. Further, their reasoning capabilities are superficial; they are not capable of making use of all the available information in the problem context. We have already initiated a project[Raghavan87e] to develop personalities that are based on formal models of inquiry and role playing, and employ expert critiquing techniques.

As the decision problem becomes large, the number of pending tasks tend to explode. We are becoming aware that our present mechanisms for task selection and scheduling are inadequate to deal with this problem. We need better approaches and criteria for deciding what part of the problem should receive attention at a given point in time, and when and how the focus should be switched.

At present, we have taken a very simple approach to interruption and resumption. The system is capable of restoring its context, and starting from where it left off. But the system provides no help to the user to regain his mental context when he resumes after interruption. In this regard, we need good mechanisms for summarizing, and explaining what has transcribed over time during a decision making process.

The philosophy and pragmatics of man-machine joint cognition dictate that knowledge and expertise will be distributed in the man-machine setup. As a consequence, the system has to acquire the necessary problem-specific and background knowledge from the user. The primary mechanism for the system to fulfill this goal is to pose questions to the user. We are discovering that this can trigger endless series of questions and can be a severe distraction from the central purpose of dialog between the user and the system. To address this problem we need systematic and effective approaches for: pruning down the questions; distributing/inserting the questions during the course of the dialog; motivating the user to cooperate with the system for satisfying the system's needs; and determining what knowledge the system should begin with, and how it should acquire additional knowledge over time.

We are also recognizing the need for the elicitation process to be 'interesting', and 'natural' to be
effective. These are unique problems of intellectual support systems. Whereas in traditional systems the consistency and predictability of the interface is a key requirement, in intellectual support systems they can be very ineffective. An intellectual support system cannot be effective if it repeats the same phrases and uses the same examples. It needs to exhibit interesting variations, have elements of surprise and unpredictability. These require the system to carry a rich knowledge base of phrases and examples, and carry on conversations in a manner that reduces monotony, and avoids repetitions.

We are also recognizing that a user may not relate to the system's use of the structuring primitives such as 'goals', 'factors', 'judgments' etc. Therefore we need to make extensive use of analogies to establish common basis for communication. We have done some initial exploration of this idea in the DRONA system [Raghavan87b].

Thus we have identified a whole set of detailed conceptual and implementation problems. These will be the focus of our on-going research. However, in the immediate future our focus will be mostly on exploring the concept of machine-based personalities in more depth.

6. Summary and Conclusions

We set out with the two goals: describing our on-going research efforts in building advanced decision support systems that can engage in an active partnership with a decision maker during decision making; and describing the JANUS system and project in detail. We have accomplished both these goals.

We started with an overview of the current approaches and themes for pursuing active support. We followed this with a detailed discussion of our research framework, methodology, and architecture for developing advanced forms of process support and intellectual support. We then presented our JANUS system to show how we are implementing and evolving our ideas. We established the unique contributions of the JANUS work by relating it to the relevant prior works. We reflected on what we have learnt through our research, and presented a list of key problems that need to be addressed in the future.

We conclude this paper by summarizing the key ideas of our work:

- Developing and implementing the process support and intellectual support approaches
- Developing a domain-independent approach to active support
- Integrating and extending the available approaches to active support, and
- Emphasizing the use of AI research paradigms in DSS research.

REFERENCES


