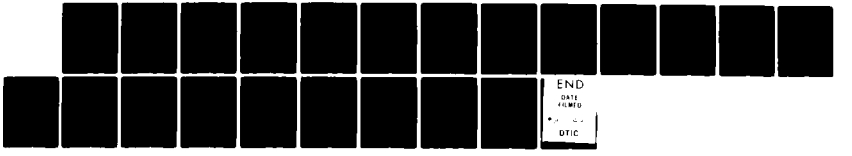
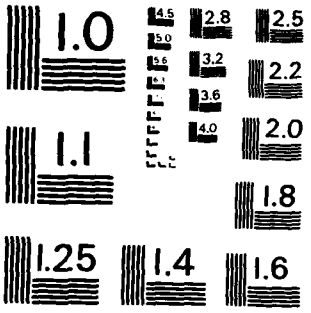
AD-A131 340 AN APPROACH TO EXPERT SYSTEMS FOR MECHANICAL DESIGN(U) 1/1
OHIO STATE UNIV COLUMBUS DEPT OF COMPUTER AND
INFORMATION SCIENCE D C BROWN ET AL. MAY 83
UNCLASSIFIED AFOSR-TR-83-0657 AFOSR-82-0255 F/G 9/2 NL





SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

REPORT DOCUMENTATION PAGE

READ INSTRUCTIONS BEFORE COMPLETING FORM

1. REPORT NUMBER: AFOSR-TR- 83-0657	2. GOVT ACCESSION NO.: A131340
3. RECIPIENT'S CATALOG NUMBER:	
4. TITLE (and Subtitle): "AN APPROACH TO EXPERT SYSTEMS FOR MECHANICAL DESIGN"	5. TYPE OF REPORT & PERIOD COVERED: TECHNICAL
	6. PERFORMING ORG. REPORT NUMBER:
7. AUTHOR(s): David C. Brown, B. Chandrasekaran	8. CONTRACT OR GRANT NUMBER(s): AFOSR-82-0255
9. PERFORMING ORGANIZATION NAME AND ADDRESS: The Ohio State University, Columbus, OH 43210	10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS: PE61102F; 2304/A2
11. CONTROLLING OFFICE NAME AND ADDRESS: AFOSR/NM, Bolling AFB DC 20332	12. REPORT DATE: MAY 1983
	13. NUMBER OF PAGES: 20
14. MONITORING AGENCY NAME & ADDRESS (if different from Controlling Office):	15. SECURITY CLASS. (of this report): UNCLASSIFIED
	15a. DECLASSIFICATION DOWNGRADING SCHEDULE:

16. DISTRIBUTION STATEMENT (of this Report)

Approved for public release; distribution unlimited.

17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)

18. SUPPLEMENTARY NOTES

IEEE COMPUTER SOCIETY, TRENDS & APPLICATIONS CONFERENCE, May 1983, GAITHERSBURG, MD

19. KEY WORDS (Continue on reverse side if necessary and identify by block number)

20. ABSTRACT (Continue on reverse side if necessary and identify by block number)

We present an approach to expert systems for mechanical design called Design Refinement, which addresses a subset of design activiety by using a hierarchy of conceptual specialists that solve the design problem in a distributed manner, top-down, choosing from sets of design plans and refining the design at each level of the hierarchy.

DD FORM 1 JAN 73 1473

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

IEEE Computer Society, Trends & Applications Conf., May 1983, Gaithersburg, MD.

AN APPROACH TO EXPERT SYSTEMS FOR MECHANICAL DESIGN

David C. Brown*
B. Chandrasekaran
Artificial Intelligence Group
Dept. of Computer and Information Science
The Ohio State University
Columbus, OH 43210

Abstract

We present an approach to expert systems for mechanical design called Design Refinement, which addresses a subset of design activity by using a hierarchy of conceptual specialists that solve the design problem in a distributed manner, top-down, choosing from sets of design plans and refining the design at each level of the hierarchy.

1. Introduction

In terms of economic impact, one of the most important areas for AI technology is CAD/CAM. AI is applicable to a variety of subtasks in CAD/CAM: process control and robotics are areas where work has already been done. However, in terms of intellectual difficulty the central problem is design itself. While much AI-related work is being done in the creation of design aid- in the VLSI area [9,10,11], there has been relatively little attention paid to the knowledge structuring and problem solving issues in the main problem of design itself. This paper addresses the problem of expert systems for mechanical design. For an important class of design tasks, we present an approach with design refinement as the central problem solving activity. This activity can be quite complex, but our aim here is to provide a first-cut analysis of this process in order to show it's potential for generating design expert systems.

---

*Currently on leave from the Department of Computer Science, Worcester Polytechnic Institute, Worcester, MA 01609

The creation of computer-based expert consultants in any area of human endeavour requires an analysis both of the knowledge structures that the corresponding human specialist uses, and the problem-solving methods that underlie the different tasks. Thus much of our discussion will be taken up with an account of these aspects of the design process.

2. Types of problem solving

We have been developing a framework for decomposing a complex body of knowledge into small knowledge sources, and organizing them into a structure of specialists engaged in collective problem solving. We have identified several distinct types of problem solving that are useful in the design of expert systems [3]. One advantage of this is that it enables one to characterize which kinds of expert problem solving we know how to mechanize.

One type of problem solving is capable of performing diagnosis, i.e., capable of reasoning about how to classify a complex description of reality as a node in a diagnostic hierarchy [1]. Another type of problem solving (called WWHI by us) is capable of reasoning about consequences of contemplated actions on complex systems, such as answering the question, "What will happen if I close that valve in this power plant?". We believe that design can be classified as a distinct type of problem solving, and that it is different from the other types we have identified. We will outline how a community of design specialists can work together to convert a list of specifications for a component into a detailed design for that component.

3. Discussion of design activity in general

In general, design is a highly creative activity, the underpinnings of which we in AI only dimly perceive. Often the design goals themselves are only vaguely specified, and the feedback from attempts to achieve these goals serves to modify them. Thus designers of new systems work with knowledge structures and problem solving techniques that we cannot yet adequately capture with AI technology. What the current state of the art in

AI can do for them is more along the lines of support systems (intelligent graphic aids, knowledgeable data bases, etc.), rather than actually performing the design task itself [13,5].

In a typical industrial operation, however, the daily task of the designer is frequently less exalted than the kind of highly creative design mentioned above. In fact, most industries perform design tasks which, for the purposes of the current discussion, can be classified into roughly three categories (to simplify what really is a spectrum from the most open ended to the most routine).

Class 1 design is done so rarely it is often the basis of a new company, division or a major marketing effort. Major inventions belong to this category. The design activity in this case involves access to wide-ranging knowledge structures, and searches in a very large space of design alternatives.

Class 2 design is closer to the routine, but still many of the established patterns may be broken. Some aspects of design may require substantial innovation; e.g., in a company which manufactures integrated sensor systems for control of sheet processes, the need to take into account extremely hot ambient conditions for a particular order may require suspension of the standard design and launching of an investigation about new (for the company) techniques of control of ambient temperature within the sensor housing, which might in turn result in new sealing techniques and so on. In many companies, this happens periodically, but is undertaken with the hope that the investment of time and energy will be paid off by identifying a potentially large market, in which the new elements of design can be "routinized".

Class 3 design, the most routine, follows a set of relatively well-established design alternatives which are reasonably well-understood, but nevertheless still require a well-trained human expert to perform the design task. We do not intend to imply that the task is "simple"; in fact, we cannot fully substitute for the human expert, and new advances in AI are called for. For example, simple approaches, such as use of a data-base of design parameters with associated designs, may work for some problems, but in general they will fail due to the large number of combinatorial possibilities.

Let us examine in some further detail the nature of Class 3 design tasks. There exists a large number of industries which make one-of-a-kind products, where each is of the same general class. For example, AccuRay Corporation, our collaborator, designs and delivers control systems to industries which manufacture sheet products (aluminum foil, paper, etc.). These control systems have sensors which continually monitor various properties of the sheets, and provide the signals which can be used to keep the thickness within certain bounds. However, even within one industry, no pre-constructed system can be placed within a functioning mill. Each control system is built anew from specifications that are gathered from a particular prospective installation. The control system itself is complex and consists of the sensor assembly (frame, carriage, sensors, etc.) and the complex computing system (minicomputers and software) that goes with the sensor hardware.

The complexity of the task arises from the numerous subcomponents and their sub-subcomponents, each of which needs to be specified according to top-level specifications. The top-level specifications of two different plants in the same industry differ considerably, and this adds to the complexity. For example, no two paper mills are alike, they themselves having been designed to the differing specifications that arise from the differing products and markets of the companies. Numerous constraints exist among the parameters of the subcomponents, contributing further to the complexity of the task.

While a Class 3 design may still be conceptually complex, the design alternatives at each stage are not as open-ended as in some of the stages of Class 2 or Class 1 designs, nor is there the vagueness and nonoperationalism of the top-level goals or the difficulty with identification of constituent substructures that is characteristic of Class 1 design. That is, despite complexity, the design is intellectually more manageable, and the variety of knowledge sources that are accessed during the execution of the task are relatively small and can be identified in advance.

Sometimes, however, a design that had been classified as Class 3 might turn out during the design process to possess many Class 2 features. This happens if the design alternatives for a certain stage all fail, and an exploratory search

for a completely new alternative needs to be launched. Identifying a design task in advance as Class 3 is a nontrivial problem. But generally, experienced designers can judge if a project is Class 3 or not.

We propose that there is a very specific type of problem solving associated with expert design activity, especially of the Class 3 type: **a hierarchy of conceptual "specialists" solve the design problem in a distributed manner, top-down, by choosing from a set of design plans and refining the design at each stage.** Each refinement is done by a specialist calling its subspecialists in the hierarchy.

4. Class 3 Design

4.1 Description of Class 3 Design

In general, the component to be designed will be quite complex, with its own subsystems and substructures. We propose that the expert system to design the component be conceived as a hierarchical collection of design specialists, where the top levels of the hierarchy are specialists in more global subsystems of the component, while those at the lower levels deal with more specific subsystems or parts. They all access a design specification data-base. Intelligent data-base assistants can play an important role here; for a discussion see [2].

At each stage a specialist S has several prioritized alternative design plans. The specialist begins by inheriting some design parameters from its parent specialist, and it obtains relevant specifications from the data-base. Each of the plans can take these data as arguments. Parts of a plan may indicate immediately that constraints cannot be satisfied. This is considered as 'failure'. Parts of a plan access functions which can fill in the design templates independently, parts produce further values or constraints to be passed on to particular successors, while other parts of a plan give specific sequences in which the successors may be invoked. Thus, S fills in some of the design, then calls its successors in a given order with requests for refinement of the design of a substructure. If

some of the substructures are independent of each other, then subspecialists may be invoked in parallel. The overall global plan of the specialist prioritizes each subplan, invokes alternate plans in case of failure by one of the subspecialists, etc. When a specialist receives failure from one or more of its successors for all its plans, or when failure for given constraints can be deduced immediately, the specialist communicates that to its parent. The exceptions to this design refinement structure are the tip node specialists who can only fill in the design details. Typically as one goes down in the hierarchy, there are fewer alternative plans, and the plans themselves becomes more straightforward.

Let us consider a concrete, but highly simplified example for illustrative purposes -- the design of a Small Table consisting of a circular Top and a cylindrical Support. In a design specification the user may specify to the design system the materials to be used, or the required dimensions. The hierarchy of specialists for the expert system to design the Small Table is shown in figure 1. Note that the hierarchy need not be a strictly component-subcomponent hierarchy, and could be organized according to function.

System Organization:

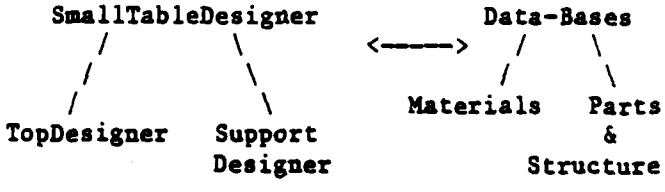


Figure 1 : Hierarchy of Specialists

The design process starts at SmallTableDesigner, at the point where the overall requirements are given to the design system. Consider the case in which SmallTableDesigner chooses its first plan, calls TopDesigner, which in turn also chooses its first plan, does the design of the Top, and returns the dimensions to its parent. Now

SmallTableDesigner calls SupportDesigner. Now suppose that this specialist's only plan fails to generate a successful design within the constraints; i.e., the strength requirement and dimension constraints are not reconcilable according to its expertise. This would cause 'failure' to be returned to SmallTableDesigner. SmallTableDesigner then calls TopDesigner again with a further constraint about the weight that is permitted. Now TopDesigner will invoke its second plan (which is a more expensive plan to execute), and some information about the new Top design is passed to SupportDesigner through its parent SmallTableDesigner, causing SupportDesigner to succeed, and the design to succeed.

An important thing to note is that very large numbers of designs are encoded in an economical way in this approach. While plans are "pre-compiled", actual designs aren't -- they are actually generated during problem solving. Further, the expertise of each specialist can be selectively increased by carefully integrating new plans into the specialist. Finally, the human designer can find causes of failure in the feedback from the expert system, and, for example, might be able to come up with a "new" way to design the support, so that the rest of the system can proceed on a more automatic design.

What makes Class 3 but not Class 2 or Class 1 amenable to this approach is the fact that in Class 3 projects, a clear idea of the identities of the specialists in the hierarchy is available (in Class 1, one doesn't even know who the successors might be for a specialist), and further, the alternative plans of each specialist can be identified and are relatively small in number (in Class 2, the needed alternative design plans are not specified).

4.2 The role of analytic routines

The image of the designer sitting in front of the CAD graphics terminal, running stress analysis routines and visually inspecting the stress patterns of a component is typical in descriptions of how CAD systems work. Analysis of design is an intrinsic part of the total design process, but what role does such analysis play in expert systems for Class 3 design?

The preceding description of how the plans work has been at too high a level to bring out this aspect. In fact, analysis of design plays a role in several steps of the process. When a specialist inherits design constraints from its parent, accesses specification data from the data base and decides if there are any obvious difficulties in proceeding with the design, one of the options will be to run some analysis packages. Similarly, at any stage in the choice of values for a design, the plan may call for some analysis. The only difference from the current CAD practice is that the analysis and evaluation will be under the control of the specialist's plans.

5. Prototype system

5.1 Introduction to Prototype System

A prototype design expert system has been implemented for the domain example used above -- that of a Small Table which consists of a circular Top and a cylindrical Support. As above, the design hierarchy consists of a Small Table specialist that uses a Top specialist and a Support specialist. The system has a small design data-base with information about dimensions of parts, the structure of the table, and the types of materials available for use. Figure 2 shows the information that a specialist has available.

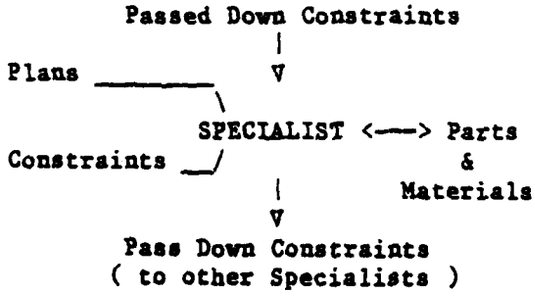


Figure 2 : Overview of Specialist

Each design specialist has a set of built-in constraints that specify what has to be true in order for that specialist to complete its design, and a collection of plans that can be selected for design, redesign, or verification of an existing design. Redesign is the alteration of an existing design due to the establishment of new specifications. As this is a class 3 task, each specialist has fixed plans that approach the design task at that level of the hierarchy in some prespecified manner. Each specialist has a 'plan suggester' that selects the plan to be executed.

The system is started by giving the Small Table designer the user's design specifications -- for example, that the top be Marble. When a specialist calls a sub-specialist all relevant constraints from the user's specification are passed down. A specialist succeeds when its selected plan succeeds, and that can only happen when its built-in and passed-down constraints are satisfied. The system uses default values to do 'rough' design, and can make small refinements to those values if necessary in order to satisfy constraints. The 'trace' of the system is very readable, with checking and decisions being handled in appropriate places. The flow of control in the system is well disciplined and it is clear at each step what part of the design is being attempted and why.

5.2 Structure

Specialists The specialists in the system are represented by a list of attribute-value pairs, plus some associated programs and constraints.

```
Specialist:
Name                  -- SmallTableDesigner
SpecialistsUsed       -- (TopDesigner
                             SupportDesigner)
BuiltInConstraints    -- (STHeight STTopDiam
                      STSupportTopDiamRatio)
DesignPlans           -- (STPlan1)
ReDesignPlans         -- (STRPlan1)
VerifyPlans           -- (STVPlan1)
PlanSuggester         -- STSuggester
```

Plans The plan-suggester's job is to select the appropriate plans to be followed during this invocation of it's specialist. In general, the plan selected could depend on the requirements given to the specialist, the request made (eg. design), the history of the design being attempted, and the current state of the design. Each specialist will have at least one plan. Below we present a typical simple plan for the SmallTableDesigner. Note that the steps marked "**" indicate places where the system will fail in an unrealistic manner, but a better plan would be too complex to present here.

```
ST Plan:
 - use TopDesigner.
 - did it succeed?
     - No, then plan FAILS.
 - use SupportDesigner.
 - did it succeed?
     - No, then plan FAILS. **
 - check to see if design meets
        table design constraints,
        and the user's constraints.
 - problems?
     - Yes, then plan FAILS. **
     - if everything OK,
        then table is designed,
        and plan SUCCEEDS.
```

Constraints Three constraints from the TopDesigner are given below. They restrict the thickness of the top, the material of the top, and the weight of the top. Constraints provide a readable specification, explicit localized tests of consistency, and can be used to direct the flow of control in the hierarchy.

```
Constraints:
  TMaterialThickness
  ((Thickness Top) > (MinThickness
                          (MadeOf Top)))

  TMaterial
  ((MadeOf Top) OneOf (Values
                         (Wood Marble)))

  TWeight
```

```
((Area Top) * (Thickness Top)
 * (UnitWeight (MadeOf Top))
< 10)
```

Data-bases Each specialist has access to knowledge about parts and materials. For each value to be specified in the design some default knowledge is available. In the implementation they are functions giving either context-free or context-sensitive values.

```
Part:
  Name              -- Top
  MadeOf            -- Unknown
  Thickness         -- Unknown
  Diameter          -- Unknown
  DefaultThickness  -- DTThickness
  DefaultDiameter   -- DTDiameter
  DefaultMadeOf     -- DTMadeOf

Material:
  Name          -- Wood
  MinThickness  -- 0.40000000E-1
  UnitWeight    -- 4
```

5.3 Action of system

In this section we will present portions of the output produced by the prototype system, in order to show its action. Note that the user's responses appear after the ">> *" prompt, and that all other text is from the design expert system. The two initial constraints specify that the table top is to be less than 2 feet in diameter, and that it must be made of Marble. Having been presented with the design specification the system can proceed.

```
DESIGN SYSTEM PROTOTYPE   (March 83)
Name of most general
  design specialist is?
>> *SmallTableDesigner

Any initial constraints?
  Answer y or n or filename
>> *CONSTRAINTS.INIT
```

```
Reading constraints from file

Constraints:
  TopSize     ((Diameter Top) < 2)
  MarbleTop   ((MadeOf Top) <-- Marble)

Using specialist -- SmallTableDesigner
In Mode ---------- Design
From specialist --- TOP
In plan ---------- TOP
Passed down ------- (MarbleTop TopSize)

Testing passed-down constraints
  MarbleTop is setting Marble
         as value of MadeOf in Top
Passed-down constraints OK

Testing built-in Constraints
Built-in constraints OK

Suggesting Plan STPlan1
Executing plan STPlan1

  Start by designing the Top
```

Before selecting its design plan the SmallTableDesigner checked the constraints, and set the top's material to be marble. The plan starts by using the TopDesigner to design the top. Once in control, the TopDesigner checks the constraints, just in case immediate failure can be reported, and then proceeds to select a plan.

```
Using specialist -- TopDesigner
In Mode ---------- Design
From specialist --- SmallTableDesigner
In plan ---------- STPlan1
Passed down ------- (TopSize MarbleTop)

Testing passed-down constraints
Passed-down constraints OK

Testing built-in Constraints
Built-in constraints OK

Suggesting Plan TPlan1
Executing plan TPlan1

  Looking for unspecified values in Top
  Try reasonable values first
```

At this point the plan for the TopDesigner is being followed, and, as this is a "tip node" in the design hierarchy, it must attempt to supply the appropriate missing details of the design. The plan continues, below, by using "default" (ie. apparently reasonable) values for those dimensions of the top that are not yet specified. After that, the design of the top appears to be complete, so it is checked using the built-in and passed-down constraints. Notice that both of the user's initial constraints are relevant for the TopDesigner and, consequently, have been passed to it. Unfortunately one of the defaults chosen conflicts with one of the design requirements. That default value gets reduced by 1 inch, allowing all constraints to be satisfied , and the design to continue. It could be argued that the default values should have been chosen with the constraints 'in mind', but we feel that the method used fits better into our refinement theory, as the defaults provide a "rough" design which is subsequently refined by consideration of the constraints.

```
    Selecting 2 ft. as Top diameter
    Selecting 0.2 as Top thickness

  Now TPlan1 checks for conflicts
  Testing passed-down constraints
    Constraints failing
      TopSize:  ((Diameter Top) < 2)
      Setting 1.9166669 as
           value of Diameter in Top
  Passed-down constraints OK

  Testing built-in constraints
  Built-in constraints OK

No conflicts found by TPlan1
Leaving plan TPlan1
Reporting Success of TPlan1 and TopDesigner

State of design:
Name            -- Top
MadeOf          -- Marble
Thickness       -- 0.19999999
Diameter        -- 1.9166669
DefaultThickness -- DTThickness
DefaultDiameter  -- DTDiameter
DefaultMadeOf    -- DTMadeOf
```

```
Name              -- Support
MadeOf            -- Unknown
Length            -- Unknown
Diameter          -- Unknown
DefaultMadeOf     -- DSMadeOf
DefaultLength     -- DSLength
DefaultDiameter   -- DSDiameter
```

At this point the SmallTableDesigner's plan calls for the design of the support, and will pass control to the SupportDesigner specialist, which proceeds in much the same way as above. Notice that here the default is context-sensitive. Note too that the SupportDesigner uses the services of a module not in the design hierarchy in order to test the strength of the support. After the support has been designed, the SmallTableDesigner checks the contraints again, and, as there are no problems, the plan is completed and the design is successful.

```
Next design the support
      .
      .
Selecting Metal as Support material,
   as Top material is Marble
      .
      .
Testing strength
      .
      .
Reporting Success of STPlan1
       and SmallTableDesigner
Result of Design attempt
  (SUCCEEDS)
```

5.4 Redesign mode

Suppose that, despite the TopDesigner having checked the weight of the Top to make sure that it wasn't too heavy, the SupportDesigner is unable to design a support that is strong enough. The SmallTableDesigner will ask the TopDesigner to redesign the top given this new information. In cases such as this we suspect that the specialist involved will be able to make a judgement as to whether this is really a request for a new design,

or a minor change to the existing design. Here, the TopDesigner would make a decision whether to select a design plan other than the one which has already been tried, or to select a redesign plan. A redesign plan will keep as much of the old design as possible and will concentrate on changing only whatever is necessary to correct the problem that the other specialist is having. Each specialist must keep or have access to a record of which plans have already been tried, under what conditions, and how successful they were.

6. AccuRay Research

The design refinement ideas presented in the previous sections are being used in an ongoing project to build an expert system for a more complex and realistic class 3 design task in an industrial environment. In conjunction with AccuRay we have studied the design of a small Air-cylinder (Figure 3). The cylinder contains a piston on a rod that moves a shutter in one of AccuRay's products. Compressed air moves the piston to open the shutter, and a spring in the cylinder, acting on the piston in the opposite direction to the air pressure, closes it. The piston moves in a sealed tube which is closed at one end by the cap, and at the other by the head. The rod passes through the Head. There are about 17 parts in all, some of them "off-the-shelf", but most are manufactured at AccuRay according to their design specifications.

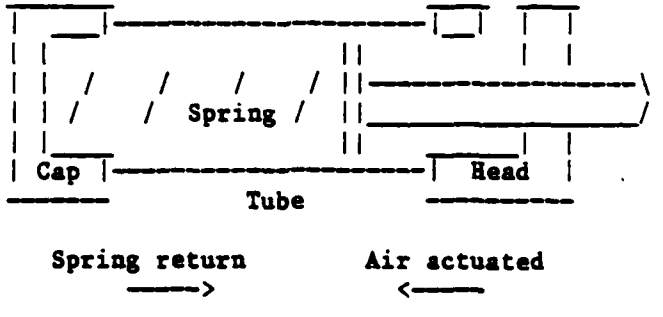


Figure 3 : Rough structure of Air-cylinder

After an extended set of interviews with the designer we have captured the 'trace' of the design process in some detail, and we are still refining it. The trace is the design decisions and their groupings considered over time. We have also isolated from this trace, in rough form, the conceptual structure and the plans that we consider underlie the design refinement process. As the implementation of the expert system progresses we expect the conceptual structures and the plans to become better defined.

7. Theoretical & Practical Issues

7.1 The language of design plans & inter-specialist communication

So far in our descriptions, plans are very general procedures. However, in order for this notion to have practical consequences in CAD we need generic representations for plans and the specification of their coordination. Otherwise, updating the expert system to reflect changing products or an increase in expertise will not be a practical. Thus, we need to search for planning primitives with which a language for design plans can be constituted. The issue of a plan specification and refinement language is especially important in our framework of problem solving types and corresponding specialist structures. We have successfully completed the task of specifying a language called CSRL for the specification of diagnostic specialists. We expect that a similar language can be designed to specify design specialists. We feel that earlier AI work on plans, such as that of Sacerdoti, Hayes-Roth and Bruce [6,7,8], will be applicable to our goal.

In our research, it is not the time sequencing of operations that is at issue, as plans have already been formed. We are concerned with the notion of constraint propagation from plans to subplans, either directly or via some blackboard [12]. However, each plan must show the sequence of tasks within that plan, some of which will use the expertise of other specialists to complete the task, and some of which will use a "compiled" [4] procedure to complete the task. Some specialists will be able to proceed in parallel.

Intimately bound up with the language of design plans is the nature of the communication between specialists. Most communication will be between a specialist and subspecialists (ie. the ones at the next lower level of the hierarchy, from which it is able to request action). The specialists may be asked to design or redesign, and could be asked to validate some small change that might affect them. Each message type will have some information associated with it. For example, the message requesting the redesign discussed above will require some information about the cause of the other specialist's failure, and, possibly, some suggestions from that specialist, or a "boss", about how to proceed.

7.2 How to handle failure

We have only scratched the surface of how failures of refinement result in reinvocation of portions of different plans. The issue is significantly more complicated. Sometimes when a plan failure occurs, it may be more beneficial to ask the parent specialist for some possibly minor changes in specifications rather than invoke alternative plans. As another example, consider the case where Plan 1 of a specialist S is being refined, and previous experience shows that a potential conflict in a specialist S' several levels below may be a most likely cause for failure of the plan. Let us also assume that several successors of S also have substantial responsibilities in the refinement of Plan 1 of S. Now it would seem prudent to selectively refine in the direction of S' to make sure early on that Plan 1 has a good chance of survival rather than engage all the relevant successors of S immediately. Finally, an important problem is how reasons for failure will be used by higher level specialists to choose alternate plans. Some degree of "understanding" the cause of failure will be necessary. At the very least some sort of classification of the causes of failure into categories that can be mapped into criteria for the selection of alternate plans will be necessary.

The above examples indicate that coordination of plans may become quite complex. Further research is called for concerning the trade-offs between overly complex plans that may capture some minor detail of the design process and sticking with

simpler plans that capture the essence of the design process, but perhaps lose some efficiency due to their incompleteness.

8. Discussion

Due to space limitations, we have not addressed several issues of interest, e.g., the conceptually important but common technique of "rough design" followed by a more detailed design based on some of the knowledge gained during the rough design phase, and the practically important problem of how to incorporate manufacturability constraints in the design process. Further it is likely that only a subset of practical industrial Class 3 problems can be successfully conquered by the design refinement paradigm. For some design tasks, we may have an insufficient understanding of the problem solving processes, or have difficulties with the amount of knowledge required. Nevertheless it is our belief that there is a significant subset of Class 3 design problems that are amenable to the proposed approach. The approach itself we think reflects in a natural manner the formation of conceptual structures for problem solving. Finally, it ought to be pointed out that while we have been mostly discussing the prospect of "automation" of design, the approach is also highly suited to semi-automation. An interactive system, in which the system, when faced with subtle issues concerning causes of failures of some designs, seeks human intervention at appropriate points in the plan selection process, will obviously be very useful. The knowledge decomposition principles that underlie our approach make the design of such semi-automatic systems particularly promising. When knowledge is decomposed into specialists, there is no particular constraint regarding which specialists need to be machine-implemented, and which can be given to human specialists.

9. References

[1]. Gomez, F., and Chandrasekaran, B., "Knowledge Organization and Distribution for Medical Diagnosis", IEEE Trans. Syst. Man. Cybern., SMC-11, 1981, pp. 34-42.

[2]. Mittal, S., and Chandrasekaran, B., "A Conceptual Representation of Patient Databases", Journal of Medical Systems, 4:2, 1980, pp. 169-185.

[3]. Chandrasekaran, B., "Towards a Taxonomy of Problem Solving Types", The AI Magazine, AAAI, Vol.4, No.1, pp.9-17, 1983.

[4] Chandrasekaran, B., and Mittal, S., "Deep Versus Compiled Knowledge Approaches to Diagnosic Problem-Solving", in Proc. Second National AI Conference, AAAI, Pittsburgh, Pennsylvania, 1982.

[5]. Fischer, G., and Bocker, H-D., "The Nature of Design Processes and How Computer Systems Can Support Them", European Conf. on Integrated Interactive Computive Systems, Stresa, Italy, Sept. 1982.

[6]. Sacerdoti, E., A Structure for Plans and Behavior, Elsevier, New York, 1977.

[7]. Hayes-Roth, B., Human Planning Processes, Rpt.No. R-2670-ONR, Rand Corp., Santa Monica, Calif., 1980.

[8]. Bruce, B., and Newman, D., Interacting Plans, Cognitive Science 2, p.195, 1978.

[9] Grinberg, M.R., A knowledge based design system for digital electronics, Proc. 1st Ann. Nat. Conf. on AI, p.283, Aug. 1980.

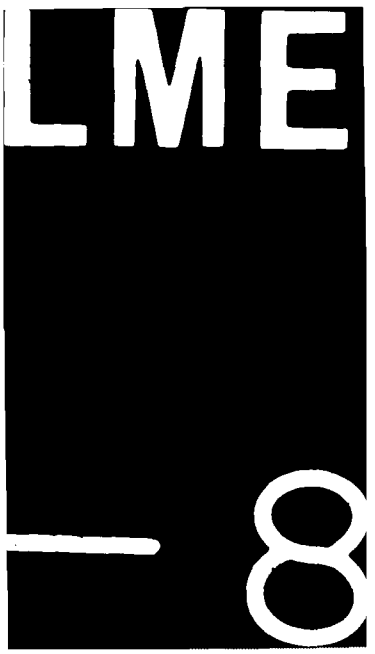
[10] Stefik, M. & Conway, L., Towards the principled engineering of knowledge, The AI Magazine, AAAI, Vol.3, No.3, p.4, 1982.

[11] Mitchell, T. et al, Representations for reasoning about digital circuits, Proc. 7th IJCAI, p.343, Aug 1981.

[12] Erman, L.D., A multi-level organization for problem-solving using many diverse cooperating sources of knowledge, Proc. 4th IJCAI, p.483, 1975.

[13] Latombe, J-C., (Ed), Artificial Intelligence and Pattern Recognition in CAD, North-Holland, 1978.

Acknowledgement: This work was supported by AFOSR grant #82-0255. We would also like to acknowledge the cooperation of the AccuRay Corporation and Pete Schmitz.


LME
8