NASA Technical Memorandum 101558

AN INTELLIGENT ADVISOR FOR THE DESIGN MANAGER

James L. Rogers and Sharon L. Padula

February 1989

National Aeronautics and Space Administration

Langley Research Center
Hampton, Virginia 23665-5225

An Intelligent Advisor for the Design Manager

J. Rogers and S. Padula

NASA Langley Research Center, Hampton, VA, USA

ABSTRACT

A design problem is viewed as a complex system divisible into modules. Before the design of a complex system can begin, much time and money are spent in determining the couplings among modules and the presence of iterative loops. This is important because the design manager must know how to group the modules into subsystems and how to assign subsystems to design teams so that changes in one subsystem will have predictable effects on other subsystems. Determining these subsystems is not an easy, straightforward process and often important couplings are overlooked. Moreover, the planning task must be repeated as new information becomes available or as the design specifications change. The purpose of this research effort is to develop a knowledge-based tool to act as an intelligent advisor for the design manager. This tool identifies the subsystems of a complex design problem, orders them into a well-structured format, and marks the couplings among the subsystems to facilitate the use multilevel tools. The tool was tested in the decomposition of the COFS (Control of Flexible Structures) mast design which has about 50 modules. This test indicated that this type of approach could lead to a substantial savings by organizing and displaying a complex problem as a sequence of subsystems easily divisible among design teams.

INTRODUCTION

Many engineering systems are large and multidisciplinary. Before the design of such complex systems can begin, much time and money are invested in determining the possible couplings among the participating subsystems and their parts. For designs based on existing concepts, like commercial aircraft design, the subsystems and their couplings are usually well-established. However, for designs based on novel concepts, like large space platforms, the determination of the subsystems, couplings, and participating disciplines is an important task. Moreover, this task must be repeated as new information becomes available or as the design specifications change. Determining the subsystems is not an easy, straightforward process and often important couplings are overlooked. The design manager must know how to divide the design work among the design teams so that changes in one subsystem will have predictable effects on other subsystems. The resulting subsystems must be ordered into a hierarchical structure before the planning documents and milestones of the design project are set. The success of a design project often depends on the wise choice of design variables, constraints, objective functions, and the partitioning of these among the design teams. Very few tools are available to aid the design manager in determining the hierarchical structure of a design problem and assist in making these decisions.

Recently Sobieski [1] showed the value of multilevel optimization as an approach to solving complex design problems. But to use this approach, a novel design problem must be decom-

posed to identify its hierarchical structure. Although much work has been done in applying AI tools and techniques to problems in different engineering disciplines (Sriram [2,3]), only recently has the application of AI tools begun to spread to the decomposition of complex design problems (Rogan [4]). Steward [5] developed a project management tool to organize and display the couplings among tasks in an NxN matrix format using matrix manipulations. A new tool has been developed to implement a decomposition scheme suitable for multilevel optimization. It displays the data in an NxN matrix format and replaces the matrix manipulations with a knowledge base to provide more flexibility. Rogers [6] presents a more detailed discussion of the tool.

This paper discusses a proposed model of the design process and describes the functions this tool uses to decompose a novel, complex design problem into a multilevel structure. The elements and functioning of the knowledge-based system , as well as a sample problem showing an application of the tool, are also presented.

A PROPOSED MODEL OF THE DESIGN PROCESS

This tool incorporates only one model of the many possible models of the design process. This model parallels Steward's [5] model of a system which defines the structure of a system as the way in which some parts of a system affect other parts of a system. These effects differentiate a system from just a collection of parts. The semantics of the system describe how and why these effects occur. The structure and semantics together completely describe the system. To attain a desirable structure, the design manager needs more formal tools to gain understanding of both the structure and the semantics of the system.

Typically, a desirable structure has a limited number of feedback links because they increase the cost of the solution. Feedback links imply that information is required before it is available which, in turn, implies that guesses must be made to initiate the process and iterations are necessary. One method of reducing feedback links is multilevel decomposition where the modules and their couplings are ordered in such a way that a number of smaller uncoupled optimization problems can be identified.

This tool partitions the modules of a system into circuits which represent subsystems where each module is simultaneously dependent on all of the other modules within the same circuit. Feedback links are contained within the circuits indicating that an iteration is required. Circuits are connected to each other only by feedforward links. This indicates that there is no iteration among circuits and they can be ordered in a multilevel format. Thus a complex design process can be decomposed into a hierarchical set of tasks.

FUNCTIONS OF THE TOOL

This tool performs several useful functions to aid the design manager in attaining a desirable structure. These functions are (1) planning, (2) scheduling, (3) displaying the modules and their couplings in an NxN matrix format, (4) displaying the subsystems in a multilevel format, and (5) displaying the dependency matrix. Each of these functions is contained in a subroutine of the main program (figure 1). The planning function is always done first followed by the scheduling function. Calling the other functions depends upon the needs of the user. After each function is completed a file is written containing the current list of modules. This allows the user to restart the process without having to go back to the start each time.

The functions of the tool are discussed in the remainder of this section using a generic design problem as a sample problem. The problem has 45 modules. These modules perform one of the following tasks: (1) set the value of one or more design variables, (2) evaluate one or more constraints functions, (3) calculate intermediate results and behavior variables, and (4) evaluate the objective function. The problem is defined in terms of relationships among these four design tasks. The dependency of the objective and constraint functions on the design and behavior vari-

ables can be defined explicitly by mathematical equations. The same is true for defining the dependency of the behavior variables on the design variables. However, the question of whether new values of the design variables can be set without knowing the outcome of the function evaluations depends on the design manager's view of the problem, therefore engineering judgement is required when determining these dependencies. The main requirement is that a design variable can only depend on a function evaluation if that function is dependent on the design variable.

Planning

The term planning within the context of this tool means determining which modules contribute to the solution of the problem. The user begins with a list of modules. This list should contain all modules that might possibly be used in the problem. The first step in the planner is to determine whether or not a module contributes to the problem by checking the output of each module against the input requirements of the other modules. If the output of the module is contained in the input list of at least one other module then that module contributes to the solution of the problem. If a module is found not to be a contributor then it is removed from the list of modules, but saved for possible use later.

In the second step, the planner examines the input lists of all the modules to determine if all input requirements are satisfied by the output of other modules. Some modules have no input requirements. These modules are used for initialization purposes and represent external inputs. If an input requirement to a module is not satisfied, then the user must interactively add a new module to the list or remove the input requirement. If a new module is added, its input requirements are also checked. If one or more of its input requirements are not met, then the modules removed from the list earlier are checked first to determine if they satisfy the requirement, if not, then another module must be added. This step continues until all input requirements are satisfied.

Scheduling

The scheduling function is the heart of this tool. Within the context of this tool, scheduling means the ordering of the modules into a meaningful solution sequence while limiting the number of feedback links among the modules. The scheduling function reorders the modules based on their couplings. If the modules and their couplings are placed into the matrix without any regard to their ordering, then very little information regarding the desirable structure of the system is available to the design manager. For example, the couplings of the initial data for the sample problem are very disorganized and contain a substantial number of feedback links (figure 2). Limiting the feedback links among the modules is done by examining the couplings and grouping the modules into circuits. This tool also orders the modules within the circuits and orders the circuits within the design process. While Steward [5] implements the grouping into circuits with matrix manipulations, this tool follows the same steps but replaces the matrix manipulations for grouping by applying rules contained in a knowledge base.

One of the advantages of using a knowledge-based tool over matrix manipulations is the ease with which new rules can be added. This gives the knowledge-based tool more flexibility. Additional rules that were not in Steward's [5] procedure were developed in conjunction with Padula's [7] design problem and have been added to control the ordering of the modules within circuits and the ordering of circuits within the design process. The ordering is done based on the weight assigned to the modules. This step reorders the modules within a circuit by moving the modules with the highest weight to the beginning of the circuit. The modules with ever decreasing weights are moved to be below but near the top priority modules to which they are coupled. Using this method, tasks can begin as soon as possible but the modules with the highest weights are given priority.

The NxN matrix display

Once the scheduling function is completed, the design manager can examine the NxN matrix display (see figure 3) and manipulate the modules and couplings to meet the requirements and semantics of the problem. This tool's display of the modules and their couplings is slightly different from that of Steward [5]. The modules of the problem are placed on the diagonal of the matrix. The couplings are lines connected horizontally to a box to indicate an output from that module and vertically to indicate an input. A circle a the juncture of the horizontal and vertical lines indicates a coupling between two modules. A circle below the diagonal indicates a feedback link.

Multilevel organization

The circuits and their couplings can also be displayed in an NxN matrix format (figure 4). By examining the circuits, it is apparent that there are no feedback links among the circuits, therefore there is no iteration among the circuits. The only iterations are contained within the circuits. Thus, once the circuits have been found during the scheduling function, it is simple to achieve a multilevel organization of the problem. A list of circuits is input to the knowledge base to determine the multilevel hierarchy. As circuits with satisfied input requirements are found, they are placed on a level. A circuit is placed on the level below the lowest level containing a circuit which generates input for the circuit being placed (figure 5).

Dependency matrix

Another function of the tool is to build the dependency matrix of the problem. The usefulness of this matrix is described by Barthelemy [8]. It is an ordered table that identifies the functional dependence between constraints and independent design variables. Behavior variables can be evaluated using design variables, therefore each behavior variable can be replaced by a list of independent design variables. Each constraint is examined to determine its dependency on design and behavior variables. Whenever a constraint depends on a behavior variable, the dependency of that behavior variable on the independent design variables is substituted. This produces a rectangular matrix with constraints listed along the rows and the independent variables along the columns (figure 6). An X marks the dependency. Building the dependency matrix after the planning and scheduling functions reveals dependency patterns that may prove advantageous when developing multilevel optimization algorithms.

THE KNOWLEDGE-BASED SYSTEM

CLIPS (C Language Production System [9]) is a knowledge-based system that was developed at NASA Johnson Space Center. It is written in C, performs forward chaining based on the Rete pattern matching algorithm, and has a FORTRAN interface. There are three main parts to this knowledge-based system, the facts, the rules, and the inference engine.

Facts are the basic form of data in the knowledge base and are contained in a facts-list. A fact is composed of one or more fields with each field separated by a space. A field can contain a number, a word, or a string. Facts can be asserted into the facts-list by an assert command in the calling program before the inference engine is executed.

The knowledge base also contains rules. A rule states that specific actions are to be taken if certain conditions are met. An action may be to return data to the calling program through the FORTRAN interface or assert a new fact into the facts-list. A rule executes based on the existence or non-existence of facts in the facts-list. Currently there are 156 rules divided among seven files.

The inference engine applies the knowledge (rules) to the data (facts) by pattern matching the facts in the facts-list against the conditions of the rule. The basic execution cycle begins by examining the knowledge base to determine if the conditions of any rules have been met. All rules with currently met conditions are placed on to the agenda which is essentially a push down stack.

Once the agenda is complete, the top rule is selected and its actions are executed. As a result of the action(s) of the rule, new rules may be placed on the agenda and rules already on the agenda may be removed. This cycle repeats until all rules that can execute have done so. The calling program passes control to CLIPS for execution of the inference engine and CLIPS returns control back to the calling program after all the rules have been executed.

AN APPLICATION OF THE TOOL

The tool was applied to the design of the Control of Flexible Structures (COFS) mast problem by Padula [7]. The COFS mast problem was a truss structure, attached to the shuttle, and used to study techniques for system identification and active control. The system contained about 50 modules for decomposition. For this system, a module represented calculating a system behavior variable such as the modes shapes, selecting a value for one or more design variables, or evaluating a constraint function. A diagram of the decomposed system is shown in figure 7. After the system was decomposed, it was divided among design teams such as structures and controls as indicated in the figure. The reference explains the changes made by the design manager to arrive at this particular decomposition.

SUMMARY

A tool using a knowledge-based system has been developed for decomposing complex design problems into a suitable multilevel structure based on the multilevel optimization approach. This tool requires an investment of time to generate and refine the input for each design module. This investment may not be justified for a small, well-understood problem, but should save a significant amount of money and time in organizing a new design problem where the ordering of the modules is still unknown. The decomposition of a complex design system into subsystems requires an interaction with the judgement of the design manager. This tool can aid the design manager in making decomposition decisions early in the design cycle.

This tool aids the design manager by reordering and grouping the modules based on the couplings among the modules. The modules are grouped into circuits (the subsystems) and displayed in an NxN matrix format. The feedback links, which indicate an iterative process, are limited and restricted to be within a circuit. Since there are no feedback links among the circuits, the circuits can be displayed in a multilevel format. Thus, a large amount of information is reduced to one or two displays which can easily be stored, retrieved, and modified. The design manager and leaders of the design teams are given a visual display of the design problem and the intricate couplings among the different modules so that they can see how a change in one subsystem will effect other subsystems. It also helps reduce the possibility of overlooking important couplings.

In addition to decomposing the system into subsystems, the tool examines the dependencies of the problem and creates a dependency matrix. This matrix shows the relationship among the independent design variables and the dependent objective and constraint functions.

Since the tool is based on AI knowledge base techniques, it has proven to be very flexible in adding new capabilities. Given its current capabilities, this knowledge-based tool can provide the design manager a great deal of insight when decomposing novel, complex design systems into more manageable subsystems; thereby saving considerable time and money in the total design process.

REFERENCES

1. Sobieszczanski-Sobieski, J.: "A Linear Decomposition Method for Large Optimization Problems - Blueprint for Development." NASA TM 83248, February 1982.
2. Sriram, D.: A Bibliography on Knowledge-based Expert Systems in Engineering. SIGART Newsletter, No. 89, July 1984, pp. 131-136.
3. Sriram, D.; and Joobbani, R.: Special Issues, AI in Engineering. SIGART Newsletter, No. 92, April 1985, pp. 38-144.

4. Rogan, J. E.: and Kolb, M. A.: Application of Decomposition Techniques to the Preliminary Design of a Transport Aircraft, NASA Contractor Report 178239, February 1987.
5. Steward, D. V.: Systems Analysis and Management. Petrocelli Books, Inc., New York, NY, 1981.
6. Rogers, J. L.: "A Knowledge-based Tool for Multilevel Decomposition of a Complex Design Problem." NASA TP 2903, January 1989.
7. Padula, S. L.; Sandridge, C.; Haftka, R. T.; and Walsh, J. L.: "Demonstration of Decomposition and Optimization in the Design of Experimental Space Systems." Preprints of the Second NASA/Air Force Symposium on Recent Advances in Multidisciplinary Analysis and Optimization. September 28-30, 1988, Hampton, VA, pp. 7-27.
8. Barthelemy, J.-F. M.: "Engineering Applications of Heuristic Multilevel Optimization Methods." NASA TM 101504, October 1988.
9. Riley, G.; Culbert, C.; and Savely, R. T.: "CLIPS : An Expert System Tool for Delivery and Training." Proceeding of the Third Conference on AI for Space Applications, November 1987.

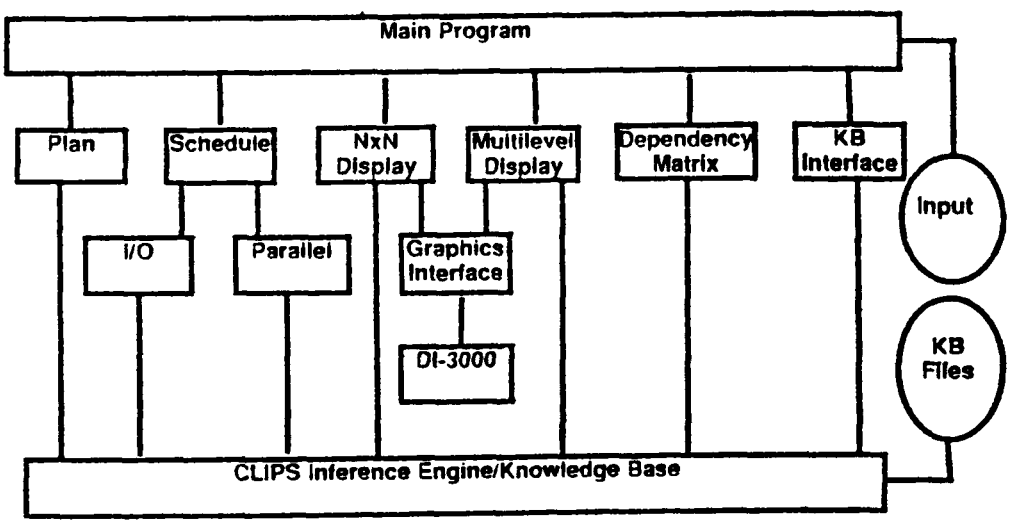

Figure 1. Diagram of the design tool.

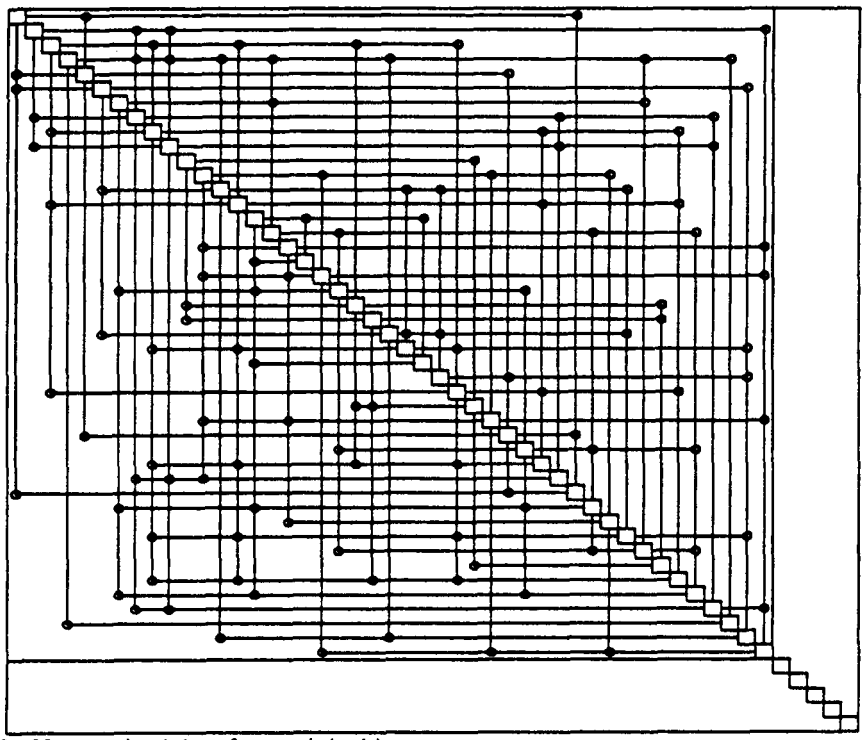

Figure 2. Unorganized data from original input.

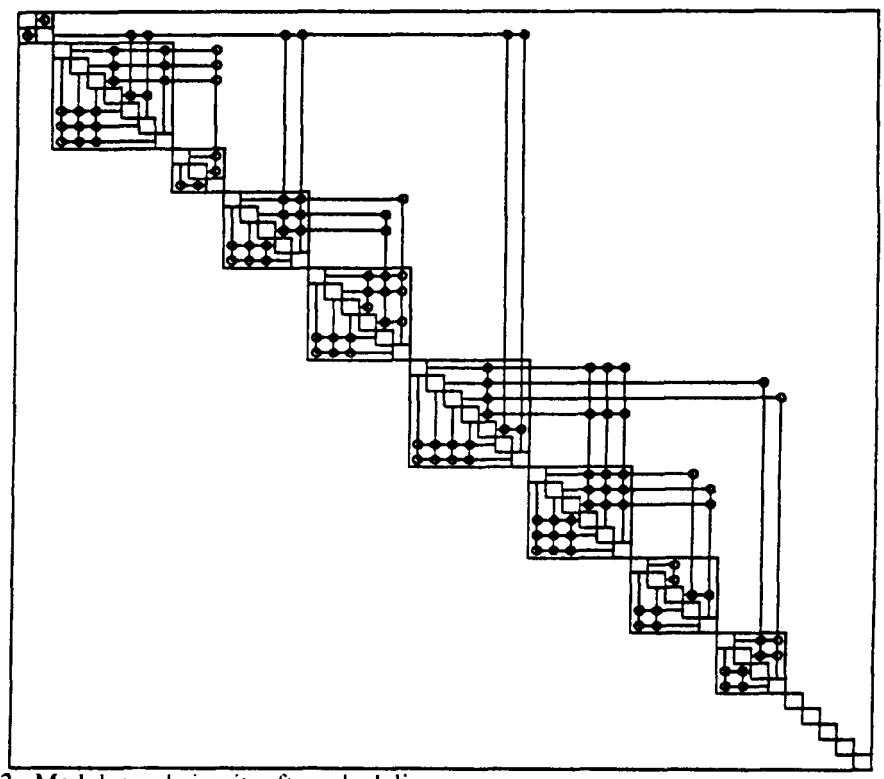

Figure 3. Modules and circuits after scheduling.

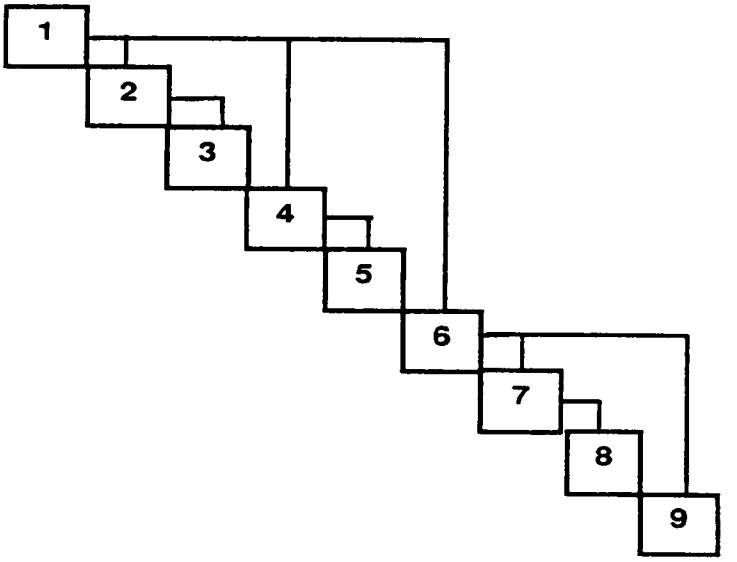

Figure 4. NxN display of circuits.

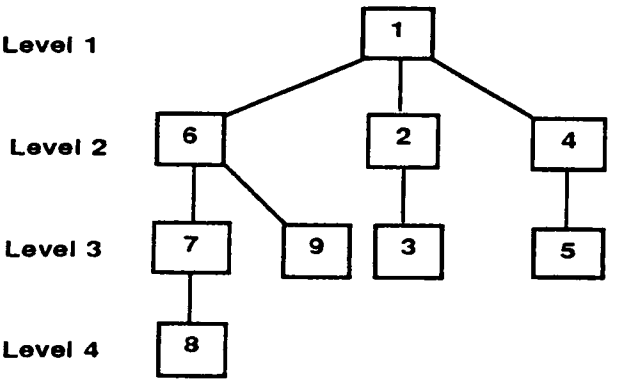

Figure 5. Multilevel display of circuits.

MOD	2	3	4	5	10	11	13	14	15	18	19	20	24	25	26	27	31	32	33	37	38	42	43
7	X	X	X	X																			
8	X	X	X	X																			
9		X	X	X																			
12		X	X	X	X	X																	
16	X						X	X	X														
17	X						X	X	X														
22								X	X	X	X	X											
23							X			X	X	X											
29	X												X	X	X	X							
30	X												X	X	X	X							
34													X			X	X	X	X				
35													X			X	X	X	X				
36													X			X	X	X	X				
40																	X			X	X		
41																		X	X	X	X		
44														X								X	X
45															X							X	X

Figure 6. Dependency matrix.

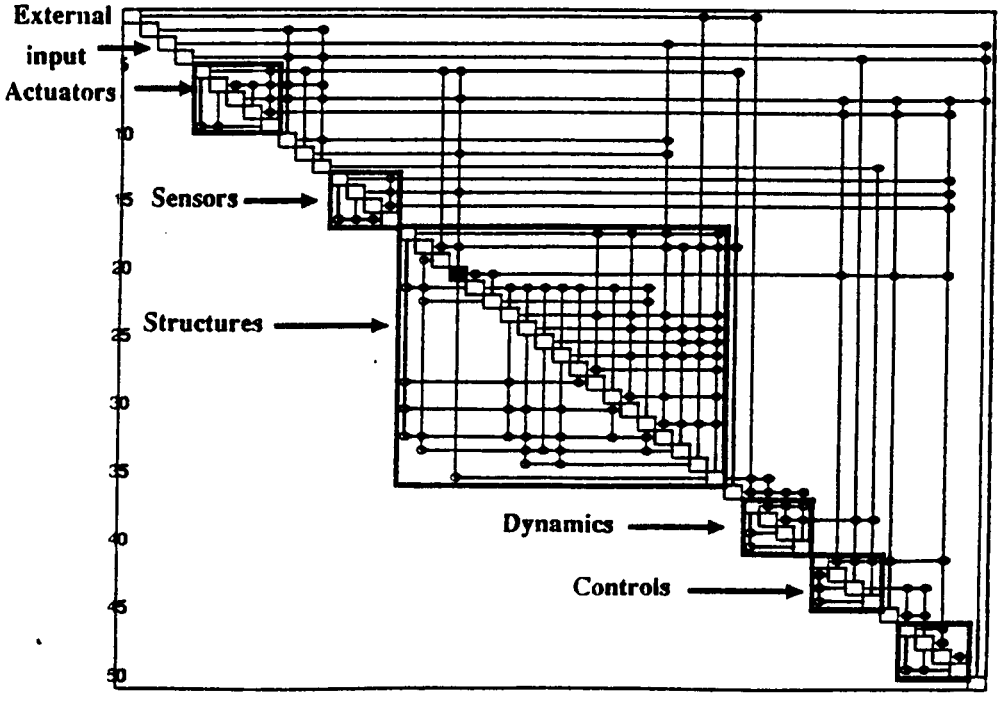

Figure 7. Desired structure for Padula [7] problem.

Report Documentation Page

1. Report No.	2. Government Accession No.	3. Recipient's Catalog No.
NASA TM-101558		

4. Title and Subtitle	5. Report Date
An Intelligent Advisor for the Design Manager	February 1989
	6. Performing Organization Code

7. Author(s)	8. Performing Organization Report No.
James L. Rogers and Sharon L. Padula	
	10. Work Unit No.
	505-63-01-07

9. Performing Organization Name and Address	11. Contract or Grant No.
NASA Langley Research Center Hampton, VA 23665-5225	
	13. Type of Report and Period Covered
	Technical Memorandum

12. Sponsoring Agency Name and Address	14. Sponsoring Agency Code
National Aeronautics and Space Administration Washington, DC 20546-0001	

15. Supplementary Notes

To be presented at the First International Conference on Computer Aided Optimum Design of Structures, Southampton, UK, June 20, 23 1989.

16. Abstract A design problem is viewed as a complex system divisible into modules. Before the design of a complex system can begin, much time and money are spent in determining the couplings among modules and the presence of iterative loops. This is important because the design manager must know how to group the modules into subsystems and how to assign subsystems to design teams so that changes in one subsystem will have predictable effects on other subsystems. Determining these subsystems is not an easy, straightforward process and often important couplings are overlooked. Moreover, the planning task must be repeated as new information becomes available or as the design specifications change. The purchase of this research effort is to develop a knowledge-based tool to act as an intelligent advisor for the design manager. This tool identifies the subsystems of a complex design problem, orders them into a well-structured format, and marks the couplings among the subsystems to facilitate the use of multilevel tools. The tool was tested in the decomposition of the COFS (Control of Flexible Structures) mast design which has about 50 modules. This test indicated that this type of approach could lead to a substantial savings by organizing and displaying a complex problem as a sequence of subsystems easily divisible among design teams.

17. Key Words (Suggested by Author(s))	18. Distribution Statement
Knowledge-base Design Decomposition	Unclassified - Unlimited Subject Category 61

19. Security Classif. (of this report)	20. Security Classif. (of this page)	21. No. of pages	22. Price
Unclassified	Unclassified	10	A02