

1989

EDESYN : an engineering design synthesis environment

Mary Lou. Maher
Carnegie Mellon University

Carnegie Mellon University. Engineering Design Research Center.

Follow this and additional works at: <http://repository.cmu.edu/cee>

This Technical Report is brought to you for free and open access by the Carnegie Institute of Technology at Research Showcase @ CMU. It has been accepted for inclusion in Department of Civil and Environmental Engineering by an authorized administrator of Research Showcase @ CMU. For more information, please contact research-showcase@andrew.cmu.edu.

NOTICE WARNING CONCERNING COPYRIGHT RESTRICTIONS:

The copyright law of the United States (title 17, U.S. Code) governs the making of photocopies or other reproductions of copyrighted material. Any copying of this document without permission of its author may be prohibited by law.

EDESYN: An Engineering Design Synthesis Environment

by

Mary Lou Maher

EDRC 12-31-89
Carnegie Mellon University

EDESYN: AN ENGINEERING DESIGN SYNTHESIS ENVIRONMENT

Mary Lou Maher

Department of Civil Engineering
Carnegie Mellon University
Pittsburgh PA 15213 USA

Preliminary design of engineering systems requires reasoning about physical subsystems, their functionalities and their combination. This part of the design process is largely qualitative, with few numerical models to guide the generation and selection of solutions. This project explores the use of knowledge-based models and techniques for representing preliminary design knowledge. The approach currently being studied is decomposition coupled with constraint directed search for feasible alternative solutions. The approach has been implemented in a software tool for developing knowledge-based expert systems for preliminary design. The tool has been applied to the domains of structural design and foundation design.

1. SYNTHESIS

Design is a process by which design intentions are transformed into design descriptions. During preliminary design, this transformation involves reasoning about the more qualitative aspects of the design as a precursor to the more quantitative aspects of engineering design and analysis. This project addresses the need for supporting the qualitative reasoning of preliminary design through the development of a shell, or a knowledge-based expert system programming environment, specifically suited for design synthesis.

Design synthesis is a part of the design process during which components and subsystems are identified and combined to form alternative design configurations. Typically, the knowledge used during this part of the design process is not well defined or articulated. Experienced designers resort to trial and error less frequently than novice designers when they synthesize designs, suggesting that the use of knowledge-based expert systems to represent "experience" may improve design synthesis.

Current design expert systems are largely domain specific systems that provide consultation on a class of design problems within a particular domain. For example, HI-RISE [1] addresses structural design of buildings, RETWALL [2] addresses design of retaining walls, V-BELT [3] addresses the design of v-belts, and VEXED [4] addresses the design of circuits.

Beyond these domain-specific expert systems for design, there is a recent interest in

understanding how to formalize the design process as information processing systems. Arciszewski [5] uses morphological analysis as a basis for generating design solutions by combining feasible states in a qualitative decisions table. In VEXED, Mitchell [4] decomposes the design process into modules and uses rules to identify the appropriate design solution for each module. In DONTE, Tong [6] represents the design process as a set of steps, where each step represents a level of abstraction and the solution is a combination of the result of each step. Mittal[7] describes a constraint oriented approach to design configuration. In DSPL, Brown [8] represents design knowledge as specialists and steps.

The common themes among all approaches cited are problem decomposition and constraint satisfaction. The approach in this project uses similar concepts in an effort to provide a domain independent environment for developing a knowledge-based system for preliminary design synthesis. The premise to this approach is that purely rule-based, forward or backward chaining representation paradigms provided by many expert system tools are not adequate for generating and comparing multiple design alternatives as is characteristic of preliminary or conceptual design.

2. EDESYN

EDESYN (Engineering DEsign SYnthesis) [9, 10] is a domain independent shell for building design expert systems. EDESYN provides a framework to define a design space to form preliminary design solutions. The design space is defined by an expert in a specific domain. The design space is organized around plans, goals, and decisions. The data concerning the initial conditions, resources, etc. is supplied by a user of the resulting expert system.

EDESYN solves a design problem by executing a plan that requires a series of goals to be satisfied. Each goal represents a decision in the design process. Thus, design solutions are formed by combining design decisions. Some salient features of EDESYN are:

1. The planning is performed during the design process. Goals are organized in the order best suited for the design problem under consideration.
2. Preliminary design is considered as the synthesis of design decisions at various levels of abstraction. During synthesis, plan generation, plan execution, and goal satisfaction techniques, combined with constraint-directed search, are applied.
3. All feasible solutions for a plan are generated. The alternatives are evaluated using heuristic criteria to identify the solution or set of solutions to be pursued further.

EDESYN consists of five main modules: synthesis processor, knowledge-base, design context, user interface, and knowledge acquisition facility, as illustrated in Figure 1. When using EDESYN, the knowledge acquisition facility is invoked first. During knowledge acquisition, the domain specific knowledge is read from files prepared by a domain expert. The domain specific knowledge is stored in the knowledge-base and the synthesis processor is invoked. The user then provides the problem specific information through the user

interface to initialize the design context and guides the synthesis of design solutions to augment the context

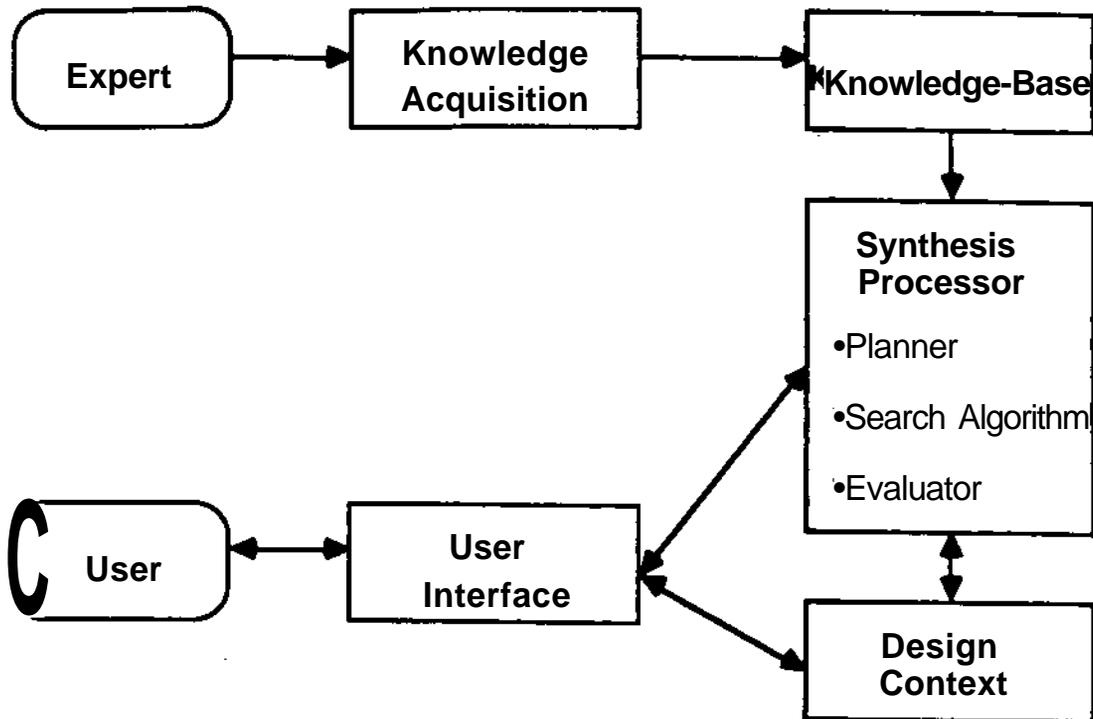


Figure 1: Architecture of EDESYN

The *knowledge-base*, developed for a class of engineering design problems, includes the following design knowledge: elements, goals, constraints, plans, planning rules, evaluation criteria, and functions.

- Elements are potential discrete solutions to the design goals. Elements may represent alternative subsystems, components, or parameters. The elements are grouped according to the goal they can satisfy. For example, the elements steel or concrete can satisfy the select-material goal. A design solution comprises a combination of elements that satisfy the design goals as well as parameters and their values.
- Goals represent decision points in the design process. A goal may be satisfied in one of three ways: select an element from a set of elements, satisfy a set of goals, or evaluate a Lisp function to return a single value. Knowledge about the feasibility requirements, represented as constraints, is associated with each goal.
- Constraints specify the functional and feasibility requirements of the artifact to be designed. The knowledge within a constraint is represented as a "condition" and an "action" part, where the condition specifies a combination of the existence of elements in the current design solution and/or the value of an initial condition, and the action specifies the element that is considered infeasible if the condition is true. Constraints in this form, i.e. elimination constraints, serve to prune the search space.

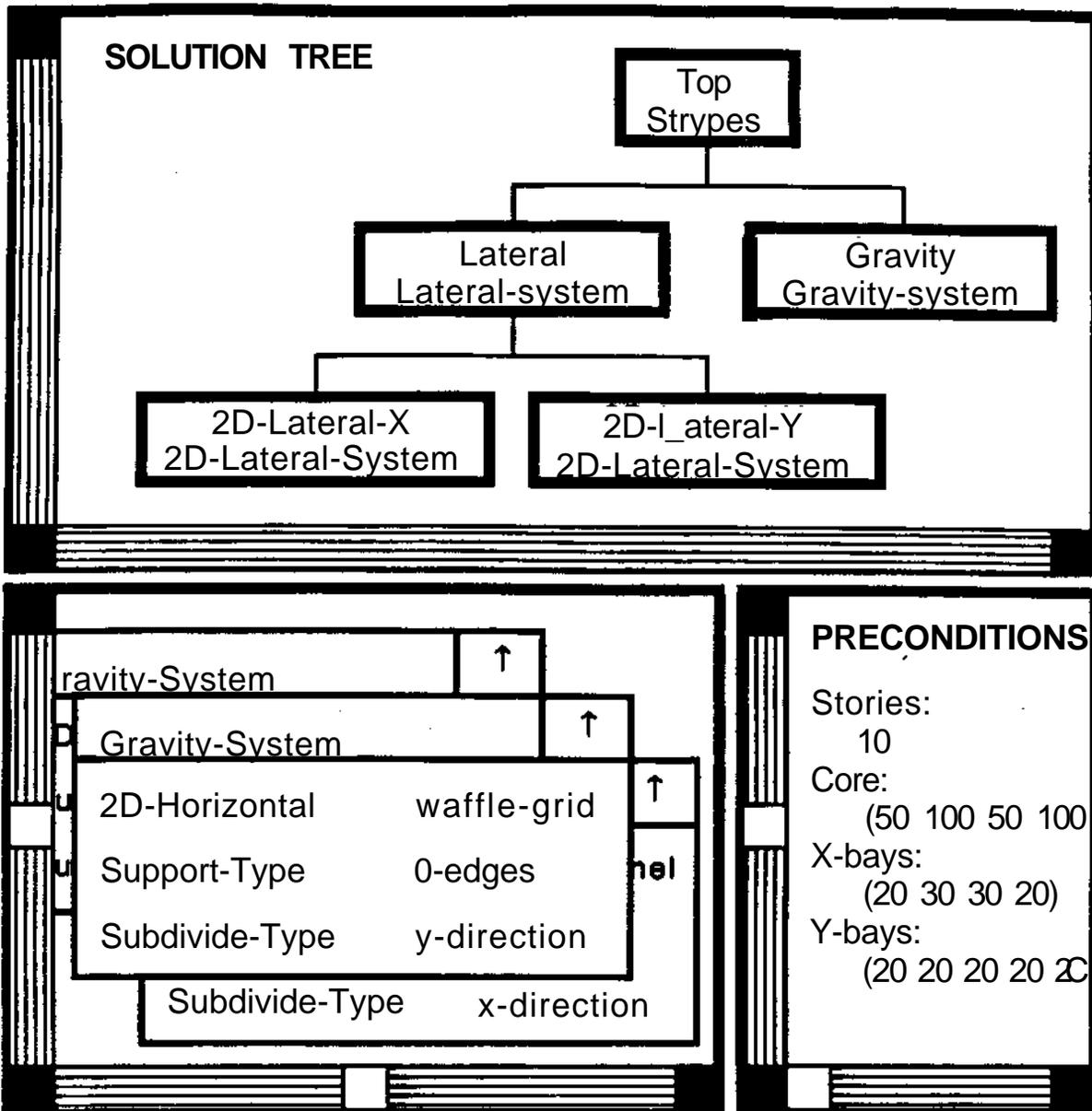


Figure 2: EDESYN's User Interface

- A plan is representative of a design problem/subproblem, and contains a set of goals to represent the problem. The knowledge about ordering these goals is also associated with each plan.
- Planning rules contain knowledge about ordering the goals in a plan. Like constraints, this knowledge is also represented in the form of a "condition" and an "action", where the condition specifies a combination of the values of initial conditions and/or the existence of elements in the current design solution, and the action specifies a list of goals that replace the default goals list if the condition evaluates to true.
- Evaluation criteria include knowledge for discriminating among feasible alternatives at a given level of abstraction. The criteria are typically based on heuristics and are assessed using partial design information. The feasible alternatives are ranked based on a weighted combination of criteria values.

- Functions contain knowledge about mathematical and symbolic expressions that can be used to calculate a design parameter. A function is connected to a goal that can be satisfied by assigning a value to a design parameter.

The *synthesis processor* consists of four components: planner, search algorithm, constraints checker, and evaluator. All four form a synthesis process to generate feasible design solutions. A planner selects and organizes the goals in an order best suited for the current design situation. A depth-first search algorithm satisfies the goals of the design problem in the order prescribed by the planner. A constraints checker checks the constraints applicable to the current goal prior to making a decision to satisfy the goal. An evaluator considers all feasible alternatives to identify a subset to be pursued further.

The *user interface* is implemented using a multi-window, menu driven interaction style. During the design synthesis process, the user can view and change the design specifications in the form of preconditions. In another window, the synthesis process is monitored by the user in the form of a tree of alternative solutions, where a path through the tree represents one solution. One solution among a list of solutions is viewed in a third window. The user interface is shown in Figure 2.

The *knowledge acquisition* facility transforms the information provided by the exper to the frame based representation of the knowledge base. The exper provides the following design knowledge: preconditions, decomposition, constraints, evaluation criteria, and functions. The design knowledge is specified in a simple syntax and stored in files. Preconditions are specified as a set of names, default values, and allowable ranges. For example, one precondition may be wind load and its default value 30 psf, and its allowable range > 0.0 . Decomposition knowledge includes the goals, subgoals, elements, and planning rules. For example the goal "design a structural system" may have two subgoals: "design a lateral load resisting system" and "design a gravity load resisting system". The constraints are specified as infeasible combinations of elements. Each evaluation criterion is specified by a name and a procedure for assigning a value using the goals and elements associated with the current solution. Functions are specified as Lisp functions that use the current state of the design solution to calculate the value of a parameter.

3. IMPLEMENTATION

EDESYN is currently implemented in Franz Lisp running on a MicroVax II. EDESYN makes use of a frame based language, FRAMEKIT [11], to define and manipulate the design knowledge and the design solutions. The synthesis processor and knowledge acquisition facility are implemented as Lisp functions. The knowledge base is implemented as related frames. The user interface facility is implemented in C using the XI1 Window Manager. The evaluator is implemented in Common Lisp as a module accessed during the constraint directed search in the synthesis processor.

The design knowledge in the knowledge base and the feasible solutions in the design context are represented in frames. The decomposition knowledge in EDESYN is represented as a tree of frames, as illustrated in Figure 3. The root of the tree is the frame representing the overall system to be designed. The slots of each frame represent (1) a relation to the parent frame, (2) subgoals for the design of the system, and (3) planning rules for selecting and ordering the subgoals. The subgoals for the design of the system are the

attributes of the system whose values are derived in one of three ways: as one-of a set of elements, the value of a Lisp function, the synthesis of a subsystem. The synthesis processor finds all feasible values for each goal in the decomposition tree, recursively applying a synthesis function to each node.

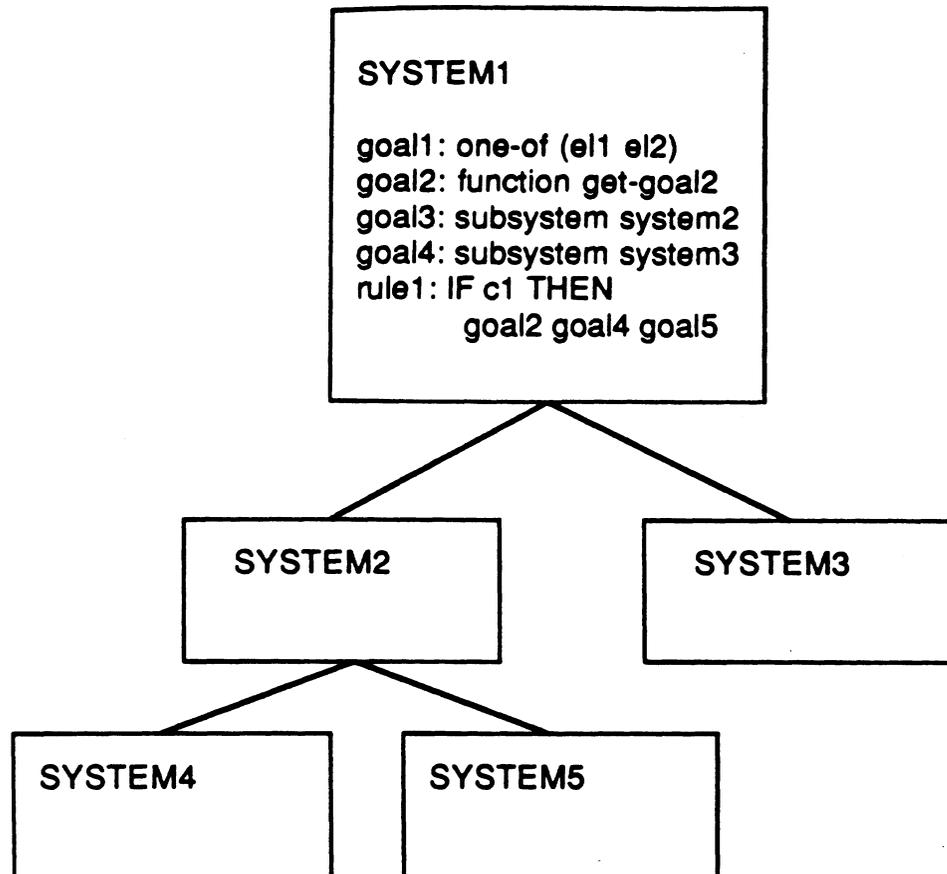


Figure 3: Decomposition Knowledge in EDESYN

EDESYN has been used as the development environment for three expert systems: STRYPES, STANLAY, and FOOTER. STRYPES generates alternative structural configurations for a building plan by identifying feasible combinations of structural types and materials. STANLAY generates feasible layouts for a given structural configuration and approximates the sizes of the structural components. FOOTER generates feasible foundation designs by identifying feasible combinations of foundation types, materials, and soil conditions. FOOTER also approximates the parameters of the foundation design alternatives. These three expert systems are used in the Integrated Building Design Environment [12].

4. CONCLUSION

EDESYN facilitates the development of design expert systems by providing a syntax for

representing design knowledge. The design knowledge is specified by the decomposition of a class of engineering systems, planning rules for applying the decomposition, constraints on recomposition, and criteria for selecting among alternatives. EDESYN follows the philosophy of expert system development by maintaining a separation of knowledge and inference mechanism. The inference mechanism in EDESYN is a constraint directed, depth-first search for feasible design solutions.

ACKNOWLEDGEMENTS

This project is supported by the Engineering Design Research Center at Carnegie Mellon University. The author would like to acknowledge James Lord for his ideas on the user interface.

REFERENCES

- [1] Maher, M.L., **HI-RISE: A Knowledge-Based Expert System For The Preliminary Structural Design of High Rise Buildings**, (PhD Thesis, Department of Civil Engineering, Carnegie Mellon University, 1984)
- [2] Hutchinson, P., **An Expert System for the Selection of Earth Retaining Structures**, (Master's Thesis, Department of Architectural Science, University of Sydney, 1985)
- [3] Dixon, J., Howe, A., Cohen, P. and Simmons, M., **DOMINIC: A Domain Independent Program for Mechanical Engineering Design**, in: Sriram, D. and Adey, R.A. (eds.), **Applications of Artificial Intelligence to Engineering Problems: Design**, (Springer Verlag, 1986)
- [4] Mitchell, T., Steinberg, L., and Schulman, J., **A Knowledge-based Approach to Design**, in: **IEEE Transactions on Pattern Analysis and Machine Intelligence**, (1985)
- [5] Arciszewski, T., **Mathematical Modeling of Morphological Analysis**, in: **Proceedings of Fifth ICMM**, (1986)
- [6] Tong, C., **Goal-Directed Planning of the Design Process**, in: **Proceedings, 3rd IEEE Conference on AI Applications**, (1987)
- [7] Mittal, S_M and Frayman, F., **COSSACK: A Constraint-based Expert System for Configuration Tasks**, in: Sriram, D. and Adey, R.A. (eds.), **Knowledge Based Expert Systems in Engineering: Planning and Design**, (Computational Mechanics Publications, 1987) 143.
- [8] Brown, D.C., **Routine Design Problem Solving**, in: Gero, J. (ed.), **Expert Systems in Engineering and Architecture**, (Addison-Wesley), in print.
- [9] Maher, M.L. and Longinos, P., **Development of an Expert System Shell for Engineering Design**, in: **International Journal of Applied Engineering Education**, (Pergamon Press, 1987)

- [10] **Maher, M.L., Engineering Design Synthesis: A Domain Independent Representation, in: (AI EDAM), (Academic Press, Vol. 1, No. 3, 1988)**
- [11] **Carbonell, J.G. and Joseph, R., FRAMEKIT +: A Knowledge Representation System, (Technical Report, Department of Computer Science, Carnegie Mellon University, Pittsburgh, Pennsylvania, 1985)**
- [12] **Fenves, S.J., Flemming, U., Hendrickson, C, Maher, M.L., Schmitt, G., An Integrated Software Environment for Building Design and Construction, in: Proceedings of the Fifth Conference on Computing in Civil Engineering, (American Society of Civil Engineers, 1988)**