ALEX: A KNOWLEDGE-BASED ARCHITECTURAL DESIGN SYSTEM

Yehuda E. Kaly
Anton C. Barfmann
Lucien M. Sverdloff

Computer-Aided Design/Graphics Laboratory
School of Architecture and Environmental Design
State University of New York at Buffalo
Buffalo, N.Y. 14214

ABSTRACT

A methodology for the development of a knowledge-based computer-aided design system and its experimental application in the domain of single-family house design are presented.

The methodology involves integrating within a unified design environment, tools and techniques that have been independently developed in various disciplines (including knowledge representation, information management, geometric modeling, human-machine interface, and architectural design). By assuming the role of active design partners, the resulting systems are expected to increase the productivity of designers, improve the quality of their products, and reduce cost and lead time of the design process as a whole.

ALEX (Architecture Learning Expert), a particular application of this methodology, is a prototype knowledge-based CAD system in the domain of single-family house design. It employs user-interactive, goal directed heuristic search strategies in a solution space that consists of a network of objects. Message-based change propagation techniques, guided by domain-specific knowledge, are used to ensure database integrity and well-formedness.

The significance of the methodology and its application is threefold: it furthers our knowledge of the architectural design process, explores the utilization of knowledge engineering methods in design, and serves as a prototype for developing the next generation of computer-aided architectural design systems.
RATIONAL

The need for a methodology that addresses issues of design efficiency and product quality stems from the increasing complexity of modern human environments, rising quality standards, and the concern for efficient use of natural resources. Conscious of these needs, researchers in architecture and other design disciplines have attempted for the past three decades to bring various computer-based methods and techniques to the assistance of human designers. While a host of computer-aided design systems have been developed for use by architects, their impact on the architectural profession as a whole has been marginal. In fact, the overall productivity of most architectural offices that use computers has improved only slightly, if at all [1,2]. In comparison, the use of computer-aided design systems by electrical engineers has increased the complexity of integrated circuits by several orders of magnitude, while significantly reducing their design time and cost [3].

Why has CAD failed to improve architectural design practices and products, and what can be done to change the current situation? CAD’s failure can probably be attributed to the fact that over 90% of all systems installed worldwide are used as drafting tools rather than design assistants [4]. Drafting, in itself, is not an essential step in the progression of a product from concept, through design, to production; but rather it is a means of communication between various activities in the process [5]. The primary use of computers as drafting aids represents a rather limited application of their computational and analytical powers in architectural design. In contrast, the use of computer aids for the design of integrated circuits extends to automatic positioning of elements in the circuit, determination of the most efficient way to interconnect them, and simulation of the behavior of the circuit as a whole. Acting as “design assistants”, computer-aided design systems have enabled electrical engineers to increase the complexity of integrated circuits by several orders of magnitude, while significantly reducing their design time [3].

The methodology presented in this paper proposes to enhance the role of computers in architectural design by integrating the former drafting, modeling, analyzing, and synthesizing tools into an intelligent knowledge-based design environment, capable of generation, representation, and evaluation of physical design solutions. This integration can be achieved by applying knowledge engineering techniques developed researchers in Artificial Intelligence, accompanied by information management methods developed by database researchers, geometric modeling methods developed by mathematicians, and domain-specific architectural knowledge.

The proposed methodology is not synonymous with the rule-based expert system approach to design automation, under development by Gero [6,7], Landdown [8], and Furness et al [9]. Rule-based expert systems apply knowledge to facts which describe the state of the designed artifact and advance it toward a final solution state. The approach presented here
utilizes knowledge engineering in a much broader sense. It integrates search control strategies with a frame-based knowledge representation network, and incorporates high level user interaction as a means to facilitate the man-machine design dialogue.

The remainder of the discussion describes a model of design as a "goal satisfying process" and its implementation in a framework for a knowledge-based CAD system. The viability of this approach is demonstrated by its application to the design of single family houses.

METHODOLOGY

Design can be modeled as a hierarchical, non-monotonic, iterative process of making decisions for the purpose of attaining specific goals, and testing how well they abide by certain constraints. A given design problem can either be solved directly or it can be recursively decomposed into sub-problems until direct solutions are possible. The individual solutions to sub-problems must then be combined to produce a solution to the higher level problem (a process which often requires application of heuristic knowledge to resolve conflicts between incompatible constraints). Additional complications arise from the need to rely on subjective criteria for evaluation and the high degree of uncertainty in predicting the actual performance of candidate solutions.

![Figure 1: Solution of a design problem.](image)

According to this model, design is comprised of two major processes:

- Generation of a symbolically represented specific solution to the given problem or sub-problem.
- Evaluation of the expected performance of this solution under the conditions dictated by the problem.

When the expected consequences of the solution can be fully predicted, these two processes can be combined into one "algorithmic" process. In most cases, however, all the consequences of implementing a specific design solution cannot be predicted because of lack of information or
complex interactions and side effects between the various components of the solution. The design of the built environment (and many other physical artifacts) falls under such non-algorithmic characterization. It requires a two-step approach of generating a candidate solution and assessing its expected performance by means of evaluative tools. Since it is often necessary to modify and re-evaluate the solution based on feedback from its performance analysis, this generate/evaluate process must be iterated until some satisfactory (but perhaps non-optimal) termination state has been reached. The communication between the generative and the evaluative steps can be accomplished through a symbolic representation of the design solution in the form of drawings, tables, computer databases, etc. The generative phase constructs and modifies this representation, while the evaluative phase uses it as input for qualitative and quantitative assessment.

IMPLEMENTATION

Given this model of the design process, a framework for a knowledge-based computer-aided design system has been proposed (figure 2).

![Diagram of CAD system](image)

**Figure 2**: Conceptual framework of a knowledge-based CAD system

The three major components of this system are:

1. Planning.
2. Design State Representation.
3. User Interface.

The first component is responsible for directing the process from its initiation through its successful conclusion. The second component keeps track of the emerging design, maintains its integrity, and provides the
basis for the decisions made by the planning component. The third component provides the means for communication between the system and its users.

ALEX (Architecture Learning Expert) is a particular application of this framework that aids in the design of single family houses. It is a prototype system intended to verify and illustrate the methodology. This specific design domain was chosen because:

- It is a common architectural design problem which exhibits the diverse range of complexities characteristic of the non-deterministic architectural design process.
- It warrants a high degree of client-architect interaction due to its personal nature and relatively small scale.
- It has the potential to serve as a practical design tool that will increase user satisfaction of the built environment.

The following discussion will elaborate on the major components of the framework, illustrated with specific examples from ALEX.

PLANNING

The planning component, which is responsible for determining and executing a strategy for solving the design problem, is made of three sub-components:

1. A set of GOALS, each consisting of a hierarchy of sub-goals, which represent the objectives that the design process should accomplish.

2. A set of EVALUATION PROCEDURES, which determine how well a given design state accomplishes the criteria and abides by the constraints prescribed by a particular goal or sub-goal.

3. The DESIGN PROCESS CONTROLLER (DPC), which determines the next goal or sub-goal to be achieved and the method for doing so.

Goals and sub-goals:

Goals, which are formulated in terms of weighted criteria, represent the various design objectives, in different levels of abstraction. The goal hierarchy used by ALEX for the design of a single family house (figure 3) has been adapted from a model of the design process proposed by Akin [10]. In this hierarchy, SINGLE FAMILY HOUSE constitutes the root goal, and is decomposed into the three sub-goals of PROGRAMMING, SCHEMATIC DESIGN and DESIGN DEVELOPMENT, each of which is further divided.
PROGRAMMING is the process by which criteria and constraints for design of the project are developed. SCHEMATIC DESIGN consists of the generation of a concept and the successive synthesis and evaluation of basic design alternatives. DESIGN DEVELOPMENT is the detailed refinement of the schematic design leading to a finished product.

Figure 3: The SINGLE FAMILY HOUSE goal hierarchy.

Evaluation:

User-generated design solutions and heuristic system-generated solutions require both absolute and relative evaluation in order to determine their merits. Absolute evaluation determines the compliance of a candidate solution with standards and codes, and establishes the merits in one particular area of concern (e.g., energy or cost). Relative evaluation determines the overall quality of the design based on the interactions between the merits of each particular area. Stand-alone "consultants" (or "evaluators"), are employed to establish absolute merits, while the responsibility for relative evaluation is assigned to the goals themselves. This allows a set of independent evaluative procedures to be used for multiple design goals, while the relative assessments (which underlie the architectural trade-off decision-making process) can be made to suit the context of particular goals and circumstances.

Design process controller:

Given the characteristics of a design state and the current phase of the design process, the DESIGN PROCESS CONTROLLER determines which goal (or sub-goal) should be achieved next, and prescribes the means to achieve it. These may include one or a combination of the following strategies.
depending on how well the task can be defined and whether the information required is available to the system:

- User interaction is used when the task is ill-understood or as an option for "manual override".

- Algorithmic processes (as demonstrated by space allocation programs), are used when the task is very well understood.

- Heuristic processes (which rely on rule-based knowledge representation) are used when the task is not well understood but its possible products are.

When none of the above direct solution methods apply, the system employs sub-goal decomposition. The sequence of goals and sub-goals which is selected by the DPC traces a dynamic "design plan" based on the particular problem's characteristics and the current state of its solution. The sequence is determined by the knowledge-base which is associated with each goal, and by the degree of adaptability of the current design state. By evaluating the merits and drawbacks of that state against the goal criteria, the DPC can effectively employ a "means-end" analysis that decreases the differences between the candidate state and the goal, and advances the solution state toward a satisfactory termination.

Figure 4: Structure of the design process controller

To illustrate the operation of the DPC, a strategy for achieving the PROGRAMMING goal in ALEX will be described. For single family housing, PROGRAMMING is often reduced to a simple statement by the client such as "I want a 3 bedroom house for $70000." Precedent suggests that this house will be approximately 1600 square feet and have, in addition to the 3 bedrooms, 2 bathrooms, a kitchen, living room, dining room, basement, garage, and perhaps a family room. The ability to make such inferences is a critical aspect of the DPC in exhibiting expert behavior.
The basic goal parameters of 4 bedrooms and a $70000 budget established by the client are combined with the system's domain-specific knowledge-base (precedent), and guide the DPC in its attempt to achieve the PROGRAMMING goal (Figure 5), using the methods available to it. The complexity of achieving the BRIEF goal leads to exploration of its sub-goals, which include SPACES, BUDGET, and QUALITY. Since the $70000 budget still meets the criteria set earlier by the BUDGET goal and there is no conflicting information about QUALITY, the DPC selects the SPACES goal to be achieved next. This goal is, in turn, decomposed into the sub-goals of ROOMS, AREAS and FUNCTIONAL REQUIREMENTS. Again, based on comparison of existing data with the goal's knowledge-base, the ROOMS goal is selected to be achieved next. Using an algorithmic method to generate a list of rooms based on the number of bedrooms and the budget, this sub-goal can be achieved directly. This is done by first generating the minimum required rooms: kitchen, bedrooms (already done) and bathrooms. The list is then expanded to include living room, dining room, foyer, garage and family room. With the achievement of the ROOMS goal, the DPC proceeds to achieve the AREAS goal. This goal can similarly be directly achieved using a minimum area algorithm to ensure that all rooms are of adequate functional size. Heuristic rules are then applied iteratively to redistribute surplus area to approach overall optimum room sizes. This iterative, hierarchical process is continued until the PROGRAMMING goal has been achieved.

![Figure 5: The PROGRAMMING goal hierarchy](image)

The flexibility of the process is increased by consideration of user input at any time. For example, if the user later decides to reduce the budget to $60000, the criteria for the BRIEF goal must be re-satisfied by decreasing existing room sizes or eliminating non-essential rooms.
DESIGN STATE REPRESENTATION

The design state representation component is responsible for storing the symbol-structures that comprise design states and maintaining their internal consistency. It is made of two major sub-components:

1. The DATABASE, which stores the objects and the links between them.
2. A CONSISTENCY MAINTENANCE SUB-SYSTEM, which ensures the integrity of the database by propagating changes that have been applied to one object over the entire network of objects.

Database:

The database comprises the symbolic representation of OBJECTS and the RELATIONS between them (Figure 6). Each object and relation is in the form of a "frame" which includes both descriptive and functional knowledge. The descriptive knowledge is comprised of form and other properties (facts) of the object (location, material, cost, etc). The functional knowledge describes how the objects (and the links) should be manipulated, and the consequences of such manipulations (rules). Together, these two kinds of knowledge constitute an abstract data type; a term coined by developers of programming languages to describe a self-consistent unit of information.

![Object and relation frame structure](image)

Figure 6: Object and relation frame structure

Links between individual objects create a network of interrelated parts which form a whole (Figure 7). These links can be classified according to
the types of relationships they simulate: part-whole, group and
master-instance relationships.

PART-WHOLE relationship links objects in a hierarchical structure. A
window, for example, is part of a wall, which is part of a room, etc. This relationship provides a natural means for propagating locational
change information: when the parent object is relocated, so are all its
children.

GROUP relationship is a general inter-object link, binding objects that
share some common property. For example, it provides a means to express a
perpendicularity constraint between two walls in a symmetrical way, so
that when either one of the two walls is rotated the other one will be
adjusted accordingly.

MASTER-INSTANCE relationship defines classification associations, where
every object belongs to a class of objects that defines its generic
parametric characteristics in terms of data and operations. While
individual objects may have different values assigned to their parameters
(such as size, location and orientation), they share common
characteristics and the internal structure of the class (e.g. all studs,
regardless of their length, location and orientation, are parallelepiped
shapes).

Utilisation of such object-centered knowledge simplifies representation of
the designed artifact and enables the propagation of changes between
objects.

![Diagram](image)

**Figure 7: Objects and links**

Consistency maintenance sub-system:

When an object in the database is affected by some externally imposed
change, such as those initiated by the design process controller, it is
important to propagate the change to other parts of the database so that
its internal consistency and well-formedness are maintained.

In the case of an open-ended database, comprised of many different
objects, the means for propagating change information must reside with the
objects themselves rather than with an external set of operators. Each
object must encode externally-significant change parameters as "messages" that are communicated to linked objects either directly or through a communal "blackboard" which is consulted by all of them. The objects then decode the messages and act upon them if relevant to their own state.

USER INTERFACE

The complex and interactive nature of the design process necessitates an effective and friendly interface between the user and the system. To support the two major forms of architectural communication, the user interface must consist of a graphical interface and a natural language interface. This allows the designer to monitor (visually and analytically) the progression of the emerging design and affect it so as to reflect his intentions. Furthermore, the user interface enables inclusion of the designer in the generate/evaluate cycle, a feature which is essential in light of the system's inability to perform all the necessary transitions between design states on its own.

The user interface fulfills another important role by providing the conduit through which the knowledge base of the system can be augmented, modified and updated. The ability to acquire new knowledge and update existing knowledge is a necessary feature to prevent early obsolescence of the system and to improve its utility. By monitoring the actions initiated by the user, new knowledge can be incorporated in the system's knowledge base, a process which is akin to "learning" by humans. For example, a novel type of furniture that has been devised by the user could be incorporated into the system as an object frame if it conforms to certain functional constraints. Similar improvements can be incorporated in the planning component by observing the sequence users prefer in progressing through goals and sub-goals. In addition to keeping the system current, the knowledge acquisition mechanism also reflects (simulates) the nature of design as an evolving, dynamic process which constantly requires modification of knowledge and revision of methods.

SUMMARY

Architectural design, understood as problem-solving, can be modeled as a process of searching for alternative solutions that satisfy certain design criteria. This non-deterministic process is complicated by the diversity of often conflicting goals and the lack of methods for achieving an optimal solution to a given problem. Therefore, the search process for a design solution requires the generation of candidate alternatives, evaluation of their degree of adaptability to design criteria and a means of consistent representation.

The symbolic representation and manipulation powers of computers are uniquely suited for simulating this generation/evaluation design cycle and
for representing the emerging artifact. Yet, systems developed so far have failed to integrate advances made in different disciplines, resulting in fewer CAD-related improvements in design practice than anticipated.

It is proposed that the full potentials of CAD (increased productivity and quality, and reduced time and cost) can be attained by creating a knowledge-based design environment which functions as an active design partner. Such a system will integrate all areas of design into a syntactic whole, assume semantic control over this syntax and self-initiate design processes.

The methodology presented involves integrating tools and techniques that have been developed independently in different disciplines (including knowledge engineering, information management, geometric modeling, and human-machine interface) with domain-specific knowledge.

A conceptual framework for a knowledge-based CAD system has been developed. The system is comprised of planning, design state representation and user interface components. Planning directs the search for a solution to the design problem. Representation keeps track of the current state of design and maintains its integrity. User interface provides the means for communication between the system and its users and enables knowledge acquisition.

ALEX (Architecture Learning Expert) has been developed and implemented as a prototype system for the design of single family houses. Currently the system is capable of assisting the designer through the programming and schematic phases of architectural design. Development is continuing on additional aspects of the design process. ALEX is written in Prolog and Pascal and runs on a VAX-11/750 under UNIX. The descriptive programming capabilities of Prolog have been utilized for the control structure of the DPC and for the description of goal hierarchies, whereas Pascal has been used for the graphic interface and some of the algorithmic processes.

Most of the discussion in this paper, particularly that which concerns the application of the methodology to the design of single family houses, draws heavily on the discipline of architecture. By substituting the architectural knowledge used for goals and evaluators with mechanical or civil engineering knowledge, it is expected that the methodology can be utilized for the design of other artifacts, and thus be of use to designers in various disciplines.

ACKNOWLEDGMENTS

The authors are grateful to Bruce Majkowski for his help in developing this paper. The research reported in this paper was supported in part by The National Endowment for the Arts under grant #A-84-017798.
REFERENCES

1. R. Nilef (moderator), "How far have we come, how far are we going, and who will benefit from the revolution?" Round Table, in Architectural Record, May 1983, pp. 39-53.


108