December 1998

Developing a Design Expert System in KappaPC

Maurice Danaher
Edith Cowan University

Wei Lai
University of Southern Queensland

Developing a Design Expert System in KappaPC

Maurice Danaher
Computer Science Department
Edith Cowan University

Wei Lai
Department of Maths and Computing
University of Southern Queensland

Abstract

The research work presented here is concerned with the development of a knowledge based system to support preliminary design. Preliminary design is the first stage in the design of buildings during which a number of structural schemes are formulated and accessed. It is a task that requires experience, knowledge and creativity as much is unknown and ill-defined, and it involves activities and decisions that are heuristic in nature. The system is currently being developed within KappaPC, the chosen expert system toolkit, using object oriented methodology. This paper discusses the approach taken in modeling preliminary design, the KappaPC environment, and the implementation of the design model within that environment. The approach presented here could be applied in other areas of design where the problem is ill-structured with many unknowns and a variety of constraints of varying importance.

Introduction

In preliminary or conceptual design the designer's task is to propose a number of feasible schemes for the building. In this phase of design, overall structural concepts and constraints are considered and much of the work is based on experience. These constraints come from many sources and range from functional constraints to aesthetics, to others such as building regulations and cost. Some of the constraints have to be fully satisfied, e.g. structural stability, whereas others may be partially satisfied, e.g. speed of construction and cost of materials. It is a task in which much is unknown and ill-defined, and it involves activities and decisions that rely heavily on experience and judgement.

Computer programs are used extensively in structural design to give accurate answers to well-defined problems such as structural analysis and the graphical presentation of results. Preliminary design, however, has not tended to be considered as a task in which programs could serve a useful purpose. Expert system techniques though could assist in the solution of this ill-structured problem.

KappaPC

The evaluation and selection of a specific shell are important parts of expert system development projects. Use of the wrong shell could lead to major compromises or possibly even project failure. A considerable number of shells are currently available with varying features and capabilities. Studies have been reported on the methods and criteria for evaluating these shells (Stylianou, Madey & Smith 1992; Stylianou 1989; Rothenberg et al. 1987).

KappaPC was chosen as the development environment for this system. This shell is written in C and runs on a standard desktop computer. It allows applications to be developed using object oriented programming methodology. Many factors were considered during the selection process. Of particular importance was the speed of execution as the model is quite large and it can rapidly increase in size during operation. The model in KappaPC's native language can be complied into C code which can run about 3 times faster than the original. Applications are developed using KAL - Kappa Application Language (IntelliCorp 1997). KAL is an interpreted language that looks somewhat like C in syntax but it behaves in a manner similar to LISP. Every KAL statement or group of statements has a return value that is the value of the last statement executed.

In object-oriented programming the domain is modeled by structures called objects. Each object contains information describing its characteristics and information specifying what it can do. In KappaPC objects are defined as classes or instances. Classes, which can be subclasses of other classes, are general objects, whereas instances are specific. The relationships between objects can be represented by linking them together into a hierarchy. The various characteristics of an object are defined within slots. An object, building, for example, may have slots for height, material, or fire rating. The actions that an object can carry out are represented by methods. A method is a programmed procedure that when activated causes the object to perform the procedure. Methods are activated by a process known as message sending. Other methods known as monitors are linked to slots and triggered when the slots are accessed. The Before-Change monitor, for example, is automatically executed just before a new value is set in a slot.

Within a hierarchy of objects, all children of an object can inherit its slots and methods. As the hierarchy is descended classes gradually accumulate inherited slots and methods. Slots and methods can also be made local, i.e. defined within a particular class or instance only. As multiple inheritance (i.e inheritance from more than one parent object) was required for this application a function was written as KappaPC does not provide this feature. Rule based reasoning is provided with the standard inference mechanisms of forward and backward chaining. It is possible to invoke the forward engine during backward chaining and vice versa. The Inference Browser graphically displays the chaining process for both forward and backward chaining. Rules can be organized into groups called rulesets allowing one to effectively direct the chaining process.

Modeling the Design Process

It is convenient to represent the design model as a specified series of levels in a structural system hierarchy (Maher & Fenves 1985; Martini & Powell 1990; Choi & Kim 1993). In accordance with Mostow's (1987) abstract refinement model, the design starts at the abstract level and is gradually refined as the hierarchy is descended. Alternative design solutions are synthesised by considering various combinations of the components in the structural system hierarchy. This type of approach allows consideration of a large number of possible designs based on how the components of the hierarchy are defined.

In this model the components, or design levels, of the hierarchy for the vertical subsystem have been defined as shown in Figure 1. This representation of the design of a building as a specified series of levels in a hierarchy, sets out clearly the tasks to be performed. All information required to design at a level is provided at that level. This includes specifying all possible structural options, design constraints, and analysis and detailing information where appropriate. At each level of the hierarchy all possible alternative designs for that level are generated. These alternatives are then tested before alternatives at the next level are generated.

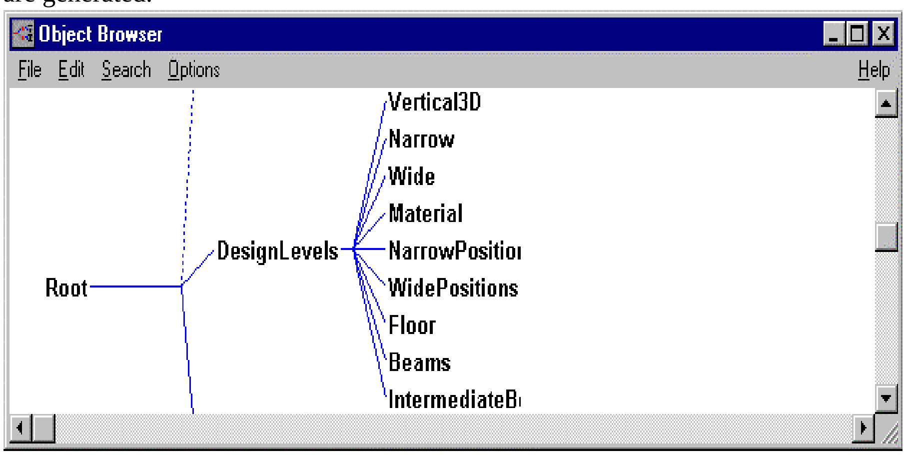

Figure 1. The Design Levels Displayed in the Object Browser in KappaPC

Implementation in KappaPC

In developing this system extensive use has been made of the features provided by KappaPC including object oriented programming, rule based reasoning, rulesets, methods, monitors, session windows and images. The design levels in the hierarchy are represented as a series of objects, each containing design information and methods appropriate for that level. All possible alternatives at each level are represented as objects, each containing pertinent design information and methods for the alternative.

The sequence of operations within the system is controlled by an object, Control. This determines the order in which operations occur, and maintains the current state of the system. The Control class has a number of slots that maintain current information such as the current task, e.g. generation of alternatives or testing alternatives, and current design level, e.g. Narrow or Material. This information is displayed to the user graphically using KappaPC's stateboxes on the session window.

The process starts by specifying details for the building to be designed. The input information includes the loading on the structure and the dimensions, e.g. the number and height of storeys, bay widths along the narrow and wide sides. Other information includes fire rating, the function of the building, the location and construction restrictions. When input is complete the Control object automatically sets the system to generation mode. The system then proceeds with the task of generating the alternative partial designs commencing at level Vertical3D.

Following the design of alternatives at each level they are tested to eliminate invalid and unlikely options. This reduces the number to be considered at the subsequent level. Without such elimination many thousands of alternative designs would be generated. The testing is performed using heuristic information, but additionally at certain levels design formulae are employed. Analysis and detailing at the last level, IntermediateBeams, prunes the remaining alternatives and produces a set of feasible designs. These remaining alternatives are structurally sound designs that are presented for evaluation.

Conclusion

Preliminary design is a suitable area for the application of artificial intelligence techniques. An approach has been presented here for developing a knowledge based expert system to assist in the task of preliminary design. Currently the system can generate a variety of alternative designs and test them using a combination of heuristic information and procedural methods.

Object oriented methodology proved to be well suited to modelling the design concept and to support the elaboration of the design during the different design stages. Systems such as this may prove to be a worthy assistant to a designer.

References

Choi, C. K. and Kim E. D., “Top Down Geometric Modeling of Buildings on Network Database,” *Computer Aided Design*, 25 (8), 468-478, 1993.

IntelliCorp, *KappaPC 2.4 Users Manual*, USA, 1997.

Maher, M. L. and Fenves, S. J., *HI-RISE: A Knowledge-Based Expert System for the Preliminary Structural Design of High Rise Buildings.* Report R-85-146, Department of Civil Engineering, Carnegie-Mellon University, Pittsburgh, USA, 1985.

Martini, K. and Powell, G. H., “Geometric Modeling Requirements for Structure Design,” *Engineering with Computers*, 6, pp. 93-1021, 1990.

Mostow, J. and Voigt, K., “Explicit Integration of Goals in Heuristic Algorithm Design,” *Proceedings of the Tenth International Joint Conference on Artificial Intelligence*, IJCAI-87, pp 1090-1096, 1987.

Rothenberg, J. et al., *Evaluating Expert System Tools: A Framework and Methodology*. Rand Publication Series R-3542-DARPA, RAND Corporation, Santa Monica, California, July 1987.

Stylianou, A. C., Madey, G. R. and Smith R. D., “Selection Criteria for Expert System Shells,” *Communications of the ACM*, Vol 35, No 10, pp. 32-48, Oct 1992.

Stylianou, A. C., *Evaluation and Selection of Expert System Shells: An Empirical Approach*. Ph.D dissertation, Department of Administrative Sciences, Kent State University, 1989.