Engineering Applications of Artificial Intelligence, Vol. 17, No. 1, 2004, pp. 11-18

Hybrid Knowledge Representation in a Blackboard KBS for Liquid Retaining Structure Design

K.W. Chau
Department of Civil & Structural Engineering, Hong Kong Polytechnic University, Hunghom, Kowloon, Hong Kong

F. Albermani
Department of Civil Engineering, University of Queensland, QLD 4072, Australia

Abstract
This paper highlights the importance of design expertise, for designing liquid retaining structures, including subjective judgments and professional experience. Design of liquid retaining structures has special features different from the others. Being more vulnerable to corrosion problem, they have stringent requirements against serviceability limit state of crack. It is the premise of the study to transferring expert knowledge in a computerized blackboard system. Hybrid knowledge representation schemes, including production rules, object-oriented programming, and procedural methods, are employed to express engineering heuristics and standard design knowledge during the development of the knowledge-based system (KBS) for design of liquid retaining structures. This approach renders it possible to take advantages of the characteristics of each method. The system can provide the user with advice on preliminary design, loading specification, optimized configuration selection and detailed design analysis of liquid retaining structure. It would be beneficial to the field of retaining structure design by focusing on the acquisition and organization of expert knowledge through the development of recent artificial intelligence technology.

Introduction
Liquid retaining structures can be classified as rectangular or circular according to shape. They can also be classified as underground or above the ground according to location. Liquid retaining structures are to be designed for hydrostatic pressure force, active earth pressure, wind force, seismic force, in addition to self-weight and imposed load. Being more vulnerable to corrosion problem, they have stringent requirements against serviceability limit state of crack. In order to select the appropriate design parameters, structural designers depend principally on their subjective judgments, intuition, professional experience, expertise, and rules of thumb. Empirical knowledge of expert designers constitutes a valuable source of information in structural design process. Yet, only a small proportion of such rules is well documented in the existing literature. It is desirable to facilitate transfer of expertise knowledge in this domain to less experienced design engineers.

Structural design is a complex task integrating both design knowledge and analytical skills. Although structural analysis in the detailed design can be automated in an algorithmic fashion, the process of structural design cannot be reduced to a predetermined mathematical sequence. Conceptual design is largely performed in the mind of the designer solely by experience and is featured by a high degree of uncertainty with often more than a unique feasible solution. However, over-emphasis has been placed on algorithmic procedures in many computer-aided design packages, thus producing a large gap between model developers and users. This may produce inferior design and cause the under-utilization, or even total failure of these models.

Design is the process to configure a system that can satisfy a variety of specifications such as resources limitations, functional requirements, and implicit and explicit design criteria. It can be regarded as a cooperative constraint-satisfying process involving a number of knowledge sources, each of which solves a sub-problem effectively by using its own specific knowledge and inferences. Each knowledge source communicates with the others to acquire information required to accomplish its tasks, or to deliver information requested by other tasks. It is an ill-structured problem, involving the selection of the best solution through minimization of either the total cost or weight of the structure and entailing skillful and iterative manipulation of knowledge. Hence it is suitable area for application of knowledge-based system (KBS).

Recently KBSs have been applied to domain problems in a variety of fields (Adeli and Al-Rijleh, 1987; Rouhani and Kangari, 1987; Chau, 1992; Chau and Yang, 1994; Chau and Zhang, 1995; Chau and Ng, 1996; Chau and Chen, 2001; Chau and Anson, 2002). Some instances of prototype systems in structural design are BTEXPERT (Adeli and Balasubramanyum, 1988), ISTRUDS (Soh and Soh, 1988), INDEX (Kumar 1995), LADOME (Lin and Albermani, 2001) and LIQSTR (Chau and Albermani, 2002). The blackboard architecture is one of the most popular systems in the implementation of KBSs in solving a wide range of tasks: speech recognition (Engelmore and Morgan, 1988), dynamic rescheduling (Bharadwaj et al., 1994), crankshaft design (Lander et al., 1996), damage assessment of steel bridge (Barai and Pandey, 2000), control of a cryogenic cooling plant (Linkens et al., 2000), large space structures (Kao and Adeli, 2002), etc.

In this paper, various knowledge representation schemes during the development of a KBS for design of liquid retaining structures are presented. It encapsulates engineering knowledge that is gathered from literature, human expert and even knowledge gleaned during the system development. All routine as well as cumbrous activities in the design cycle are covered.

Programming paradigm and framework

In order to develop an integrated design environment for liquid retaining structures using the KBS approach, a number of issues including the development platform, development tool, problem-solving strategies, knowledge representation paradigms and flexibility for future extension need to be addressed. After acquisition of the necessary knowledge, it must be transformed into a representation form amenable to programming so as to develop a computer system that emulates the working processes of a human expert. During this translation process, the system architecture has to be established.

Expert system shell

In order to facilitate the development of KBSs, expert system programming environments or shells have been developed with specific representation methods and inference mechanisms. The following factors are taken into account in selecting an expert system shell: type of machine and operating system; knowledge representation schemes; type of control strategy and inference mechanism; user interface; ability to interface with external programs; availability of complex mathematical routines; and, explanation facilities. Since knowledge does not always exist in an appropriate form ready for use, knowledge representation techniques play a key role in this aspect in order to arrange the available knowledge into a format such that the KBS can use it effectively to solve the domain problem. Its selection is based on the programming tool in the system development process and the nature of problem under consideration. Previously, rule-based, frame-based, or object-oriented knowledge representation scheme was usually employed exclusively in developing KBSs. For design task that is of formation or synthesis nature, rule-based representation is not adequate.

Object-oriented programming is more versatile because of its modularity, data abstraction and inheritance characteristics. A hybrid programming technique, which integrates the use of several knowledge representation methods as appropriate, takes advantages of the characteristics of each scheme. To this end, the development environment has to cater for this hybrid representation technique.

In this study, an expert system shell, which is a hybrid application development tool integrating object-oriented techniques, relational database models, expert system technology, and traditional procedural programming, is employed. The well-established debugging tool facilitates programming error detection. Figure 1 shows a sample screen shot of the debugging tool during coding of this system. When the source codes are being entered, it provides hints or tips on parameters for methods and functions, properties and methods of an object and a list of acceptable constants.

Blackboard architecture

Blackboard system encapsulates information sharing through the common data structure termed a blackboard, which stores the current state of the solution, problem data, intermediate parameters and final design outcomes. This architecture furnishes a problem-solving model with contribution from a multitude of knowledge sources at different levels by integration into a single system through this blackboard. Through the use of rules and frames under object-oriented programming environment, various knowledge sources are grouped into separate modules. This serves the requirement for design of an engineering structure, which is usually characterized by interaction between diversified knowledge sources.

Knowledge acquisition and representation

The knowledge base contains class declarations, backward-chaining rules, forward-chaining methods, database function, displays and interfacing facilities to external programs. The knowledge base comprises design knowledge originated from many sources, including literature, empirical data, knowledge derived during the system development, as well as heuristic knowledge by expert. A myriad of knowledge representation paradigms is adopted to tailor for each type of domain knowledge in the knowledge base.

Objects

Objects are created via class declarations, which represent declarative knowledge in a knowledge base. Similar objects are grouped into a class, which defines the structure, properties, operations and inheritance of an object. Class declarations encapsulate the structure (attributes) and behavior (facets, methods, rules, and demons) of the object and allow code sharing between objects within the same class. Figure 2 shows the objects in the knowledge base, which are classified into knowledge modules and the blackboard. Knowledge modules corresponding to procedural expertise knowledge in solving design problem is divided into Design Process and Process Control whilst objects in the blackboard are basically classified into Design Stage and Design Entities.

The blackboard is partitioned into a number of hierarchical levels, corresponding to different stages of the design process. This kind of declarative knowledge is unable to effect program execution merely by itself, but the attribute values of different objects can be stored and retrieved whenever they are required during the problem solving process. This organization emulates closely the reasoning mechanism of a human expert designer. Either one of the following attribute types, namely, compound, multi-compound, instance reference, numeric, simple, string, interval, and time, is defined for each class. A facet designs the inference strategy for processing an attribute. A search order list is set optionally for each attribute, whose value is obtained from rules, session context, default value, method or end-user query. Design Stage only comprises a single object whereas there are several objects in the Design Entities level. Data inside Design Stage are employed by the Process Control knowledge modules to determine the next possible action, or to check the validity of the function triggered by the user. Forward chaining inference mechanism is employed here to derive the next design process. After a specific design stage has been satisfied, the pertinent Design Stage indicator will be assigned one of the values from the preset value list.

Design Process modules determine largely the scope of design to be solved by the KBS. The design of liquid retaining structures is represented by eleven objects, namely, Analysis and Sizing, Crack Width Checking, Final Member Details, Imposed Load Specification, Load Combination Specification, Model Specification, Structural Specification, Alternative Evaluation, Sectional Properties Retrieval, Support Specification, and Wind Load Specification. The attached procedural method is processed when the value of the attribute changes, either by assignment under another method or by the user. Figure 3 shows the representation of the key design process by a semantic network. A mixed problem-solving strategy is used here. The user is required merely to supply the relevant data during each design stage and the system will determine the order in which different design knowledge modules are executed.

Process Control modules ensure the proper and effective application of knowledge in Design Process modules. They evaluate the current attribute values in Design Stage of the blackboard, which provides the indicator to assist this decision making. The Main Design Process class monitors the design stage of all key tasks during the design process and decides either to continue to next step or to prompt a warning message. All primary tasks in Process Control module are expressed on command buttons together with procedural methods attached. Classes such as Preliminary Design, Load Determination, Analysis and Sizing, Design Summary, and Miscellaneous are employed to control the execution of subtasks such as to enable several command buttons, to change the font color of a command button, to close one of the opened windows, to assign a new attribute value of a design stage, etc. Process Control knowledge modules work closely with the user-interface module to produce user-friendly main menu displays, which is important for a functional KBS. Moreover, the relevant entries and design parameters under Design Entities, the corresponding attribute values of Design Stage are synchronized through the Process Control knowledge modules.

Procedural methods

The design procedures are represented using procedural methods and conventional algorithmic programs interfaced with the system (British Standards Institution, 1987). Procedural knowledge expressed using methods are often represented as program codes attached to attributes. Here, methods are often attached to the command buttons or option buttons. However, procedural methods are only attached to attributes of objects in Design Process and Process Control knowledge modules representing the design processes whilst objects in the blackboard representing the design context do not embed any procedures. Figure 4 shows a typical example of procedural method attached to optional button OptCircular in the preliminary design screen.

Rules

Rules describe the operational logic and cause-and-effect relationships, which are needed to make decisions and to fire certain events or actions during execution. In general, production rules are considered an intuitive means for representation of heuristic knowledge, which mainly entail symbolic representation. If the antecedent of a rule is determined to be true, the inference engine may fire the rule, inferring the conclusion statements to be true, which is then added to the working memory. Such rules are invoked mainly through change in pattern of other subprograms, instead of through a call from other subprograms in a specified algorithmic fashion. During each cycle, the conditions of each rule are matched against the current state of domain contexts. Rules are grouped into a rule set representing a collection of production rules with the same attribute as the conclusion. The rule sets include the knowledge necessary for the determination of different material properties, various geometrical ratios, interpolation of moment and shear coefficients, and selection of design parameters such as shape factors in wind load determination. As a typical example, the rule set as shown in Figure 5 demonstrates how to determine height aspect factors. The rule codes are easily comprehensible and hence this inference network, constituted by the rule sets, can be further extended as more and more knowledge is gleaned and becomes available.

Databases
A database system is typically a record-keeping system employed to maintain relatively large amount of data. Some types of engineering knowledge are represented more conveniently in a database format. Here, database tables are used to represent engineering knowledge, such as moment and shear coefficients for various configurations in preliminary design, structural properties of reinforced concrete sections, structural properties of proposed alternatives and final member details in detailed design. Some heuristics are used to limit the choice of some design parameters to only practical values. Some of the feasible design parameters, acquired from practice engineers and code requirements, are given in Table 1. These databases contribute as a part of the entire design knowledge. Some of them such as the moment and shear coefficients are static and are not changed by any design activity whilst the others such as the database on structural properties of proposed alternative are dynamic and are generated during the execution of the system.

Traditional algorithmic codes
Most KBS development tools are not tailored for numerical processing but, instead, are designed for symbolic processing. However, this system can handle both symbolic and algorithmic programs simultaneously. Algorithmic models include preliminary design, numerical model generation, code conformance checking, optimized member sizing and finite element structural analysis. Custom-built codes as well as available existing codes are employed to perform these number-crunching tasks. An algorithmic package is adopted to perform nonlinear finite element analysis. Upon completion of execution of the external program, the previous session in the KBS is resumed. Of course, the Process Control knowledge modules continue to control the actions to be taken, depending on the outputs.

Inference engine
An inference engine controls the selection of procedure methods and production rules from the knowledge base to derive a conclusion or design context. All the design steps can be seen explicitly on the main screen display. The validity of the user’s choice on the preferred sequence of design processes is checked by Process Control knowledge modules, which act opportunistically upon being triggered. An event-driven inference processing mechanism is adopted so that the ensuing action of the system will depend on the input made by the user. For example, the applicable design loads on the structure is considered in accordance with the type of liquid retaining structure. If underground liquid retaining structure has been selected, the user is prompted to enter the soil properties. If liquid retaining structure above the ground

has been chosen, the user is prompted to enter the wind load parameters. Figure 6 depicts the flowchart showing the overall design algorithm of the prototype system.

Application example

A typical example of liquid retaining structure is employed to demonstrate the application of the system. An underground circular shape liquid retaining structure is designed under a very severe exposure environment. User-friendly displays are used to interact with end users by prompting for values and showing the output data. The sample run commences with Structural Specification in the main menu. The system checks for consistency and accuracy of all input data and prompts a warning message if the data is not within the specified range. During the preliminary design stage, heuristics are used to evaluate different alternatives. Table 2 shows an example of such heuristic design table showing the moment and tension coefficients for side wall of circular shaped liquid retaining structures with different height, diameter and thickness relationships. Based on these geometric constraints and crack width requirement, the KBS searches the databases on moment and tension coefficients and on sectional properties and proposes 15 feasible proposed configurations in order of priority of minimum costs of reinforcement and concrete. Figure 7 shows the screen displaying details of a proposed alternative. Detailed specification is followed prior to detailed design and analysis. Figure 8 shows the screen displaying the support specification. The structure is analyzed for various load combinations in accordance with the code provisions. After the iterative process of numerical model generation, structural analysis, code conformance checking and member sizing, the KBS determines the structure with minimum cost.

Conclusions

In the development of this KBS for design of liquid retaining structures, hybrid representation format to suit the characteristics of different types of knowledge under a blackboard architecture is shown to be capable of gathering different stages of the structural design processes together. The coupling of algorithmic programming and heuristic rules on the basis of a combined factual/empirical knowledge base renders improvement on computer-aided design by substantially shorten the process duration and by elimination of human errors in data transfer. The prototype system, intended to be a guide for both experienced and novice users, acts as an intelligent assistant that guides the user throughout the design process on liquid retaining structures. It is very useful since, otherwise, the user needs to refer to many codes and consult experts for an optimized design. In addition to being a valuable tool for future designers, this repository can be used as a teaching or a training tool to help students to organize their thought processes.

References

Adeli, H. and Al-Rijleh, M.M., 1987, A knowledge-based expert system for design of roof trusses, *Microcomputers in Civil Engineering*, 2(3), 179-95.

Barai, S.V. and Pandey, P.C., 2000, Integration of damage assessment paradigms of steel bridges on a blackboard architecture, *Expert System with Applications*, 19(3), 193-207.

Bharadwaj, A., Vinze, A.S. and Sen, A., 1994, Blackboard architecture for reactive scheduling, *Expert System with Applications*, 7(1), 55-65.

British Standards Institution, 1987, *BS 8007: Design of Concrete Structures for Retaining Aqueous Liquids*, British Standards Institution, London.

Chau, K.W., 1992, An expert system for the design of gravity-type vertical seawalls, *Engineering Applications of Artificial Intelligence*, 5(4), 363-367.

Chau, K.W. and Albermani, F., 2002, Expert system application on preliminary design of water retaining structures, *Expert System with Applications*, 22(2), 169-178.

Chau, K.W. and Anson, M., 2002, A knowledge-based system for construction site level facilities layout, *Lecture Notes in Artificial Intelligence*, 2358, 393-402.
Chau, K.W. and Chen, W., 2001, An example of expert system on numerical modelling system in coastal processes, *Advances in Engineering Software*, 32(9), 695-703.
Chau, K.W. and Ng, Vitus, 1996, A knowledge-based expert system for design of thrust blocks for water pipelines in Hong Kong, *Water Supply Research and Technology - Aqua*, 45(2), 96-99.
Chau, K.W. and Yang, Wen-wu, 1994, Structuring and evaluation of VP-expert based knowledge bases, *Engineering Applications of Artificial Intelligence*, 7(4), 447-454.
Chau, K.W. and Zhang, X.N., 1995, An expert system for flow routing in a river network, *Advances in Engineering Software*, 22(3), 139-146.
Engelmore, R. and Morgan, T., 1988, *Blackboard Systems*, Addison_Wesley, Wokingham.
Kao, W.M. and Adeli, H., 2002, Multitasking object-oriented blackboard model for design of large space structures, *Engineering Intelligent Systems for Electrical Engineering and Communications*, 10(1), 3-8.
Kumar, B., 1995, Knowledge Processing for Structural Design, *Topics in Engineering Volume 25*, ed. Brebbia, C.A. and Connor, J.J., Computational Mechanics, Southampton.
Lander, S.E., Corkill, D.D. and Staley, S.M., 1996, Designing integrated engineering environments: blackboard-based integration of design and analysis tools, *Concurrent Engineering Research and Applications*, 4(1), 59-71.
Lin, S. and Albermani, F., 2001, Lattice-dome design using a knowledge-based system approach, *Computer-aided Civil and Infrastructure Engineering*, 16(4), 268-286.
Linkens, D.A., Abbod, M.F., Browne, A. and Cade, N., 2000, Intelligent control of a cryogenic cooling plant based on blackboard system architecture, *ISA Transactions*, 39 (3): 327-343.
Rouhani, S. and Kangari, R., 1987, Landfill site selection: a microcomputer expert system, *Microcomputers in Civil Engineering*, 2(1), 47-53.
Soh, C.K. and Soh, A.K., 1988, Example of intelligent structural design systems, *Journal of Computing in Civil Engineering, ASCE*, 2(4), 320-345.

Design parameter	*constraints*
Minimum concrete cover	40 mm
Concrete cover	Either 40, 50, 60 or 75 mm
Minimum concrete grade	Grade 30
Concrete grade	Grade 30, 35 or 40
Maximum concrete grade	Grade 40
Minimum reinforcement grade	Mild steel
Maximum reinforcement grade	High yield steel
Minimum area of steel	0.4% of gross cross-sectional concrete area
Maximum area of steel	4% of gross cross-sectional concrete area
Minimum slab thickness	200 mm
Slab thickness	Either 200, 225, 250, 275, 300, 350, 400, 450, 500, 600, 700, 800, 900 or 1000 mm
Minimum diameter of reinforcement	10 mm
Diameter of reinforcement	Either 10, 12, 16, 20, 25, 32 or 40 mm
Maximum diameter of reinforcement	40 mm
Allowable crack width	0.1mm or 0.2mm
Minimum spacing of reinforcement	100 mm
Spacing of reinforcement	Either 100, 125, 150, 175, 200, 225, 250, 275 or 300 mm
Maximum spacing of reinforcement	300 mm

Table 1 Constraints on various design parameters in design of liquid retaining structure

H^2/Dt	*Moment Coefficient*	*Tension Coefficient*
2	0.0447	0.276
2.5	0.0381	0.329
3	0.0333	0.371
3.5	0.0294	0.407
4	0.0264	0.436
4.5	0.0239	0.462
5	0.0219	0.484
5.5	0.0203	0.502
6	0.0187	0.522
6.5	0.0176	0.536
7	0.0164	0.551
7.5	0.0154	0.564
8	0.0146	0.575
9	0.0131	0.598
10	0.012	0.617
12	0.010	0.649
14	0.0088	0.673
16	0.0079	0.693
18	0.0071	0.711
20	0.0063	0.725
25	0.0052	0.755
30	0.0043	0.776
40	0.0033	0.809
60	0.0023	0.848
80	0.0017	0.872
100	0.0014	0.889
200	0.0014	0.931
500	0.0014	0.971
1000	0.0014	0.991

Notes:

H = vertical height; D = diameter of circular tank; t = wall thickness; w = specific weight of liquid

Moment = moment coefficient x wH^3; Tension force = tension coefficient x wHD/2

Table 2. Moment and tension coefficients for side wall of circular shaped liquid retaining structure with topside free

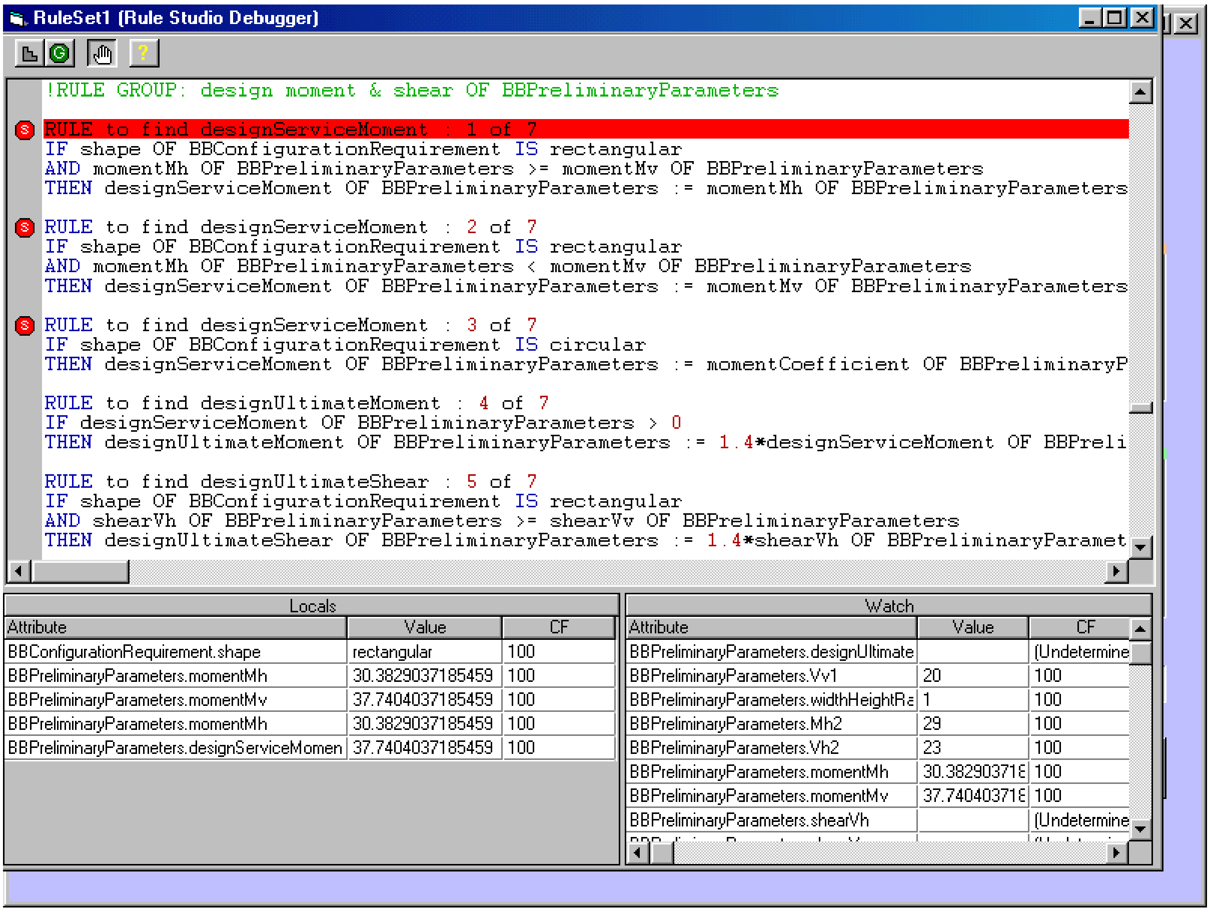

Figure 1. A sample screen shot of the debugging tool

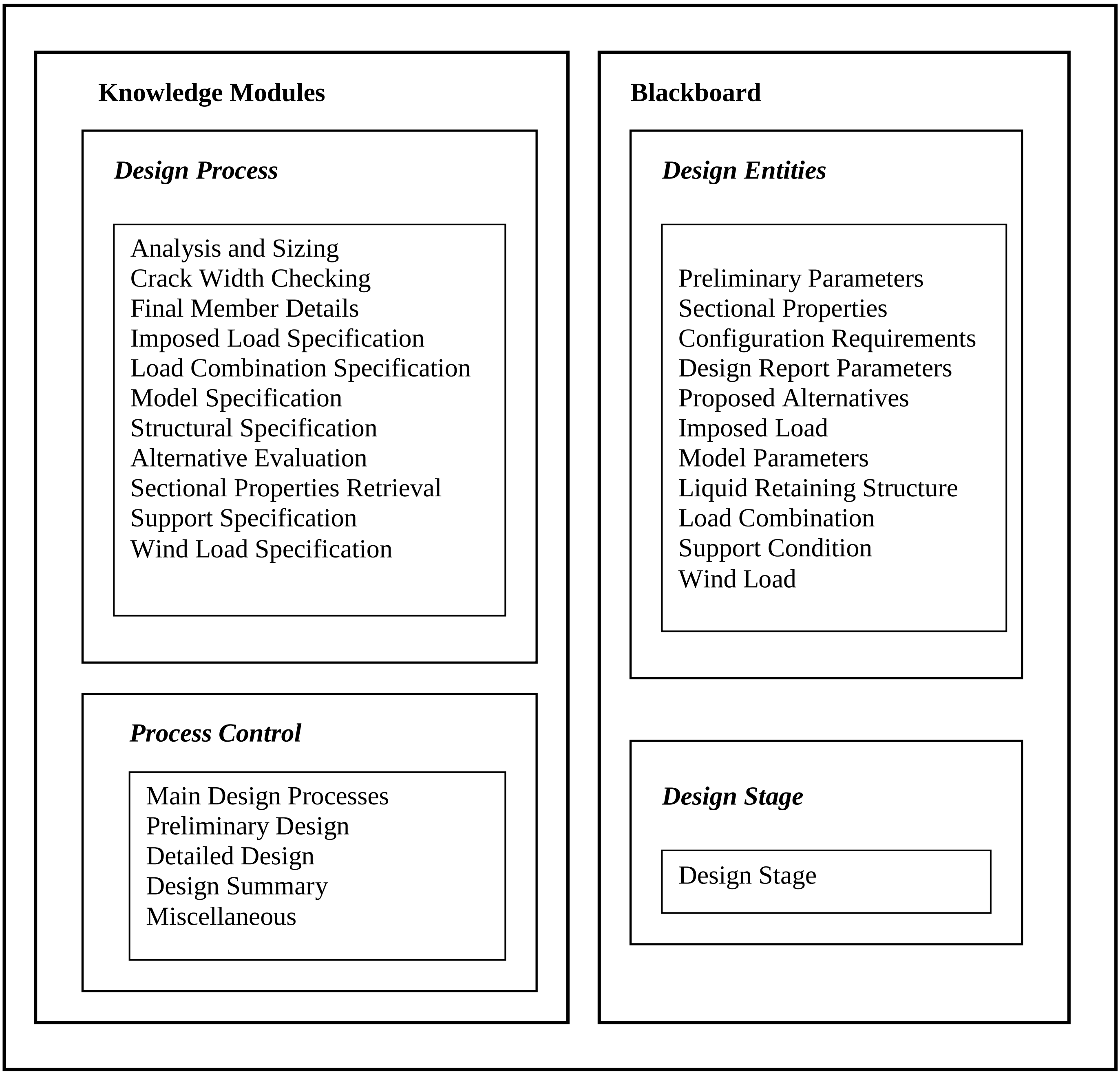

Figure 2. Various Classes in the knowledge base

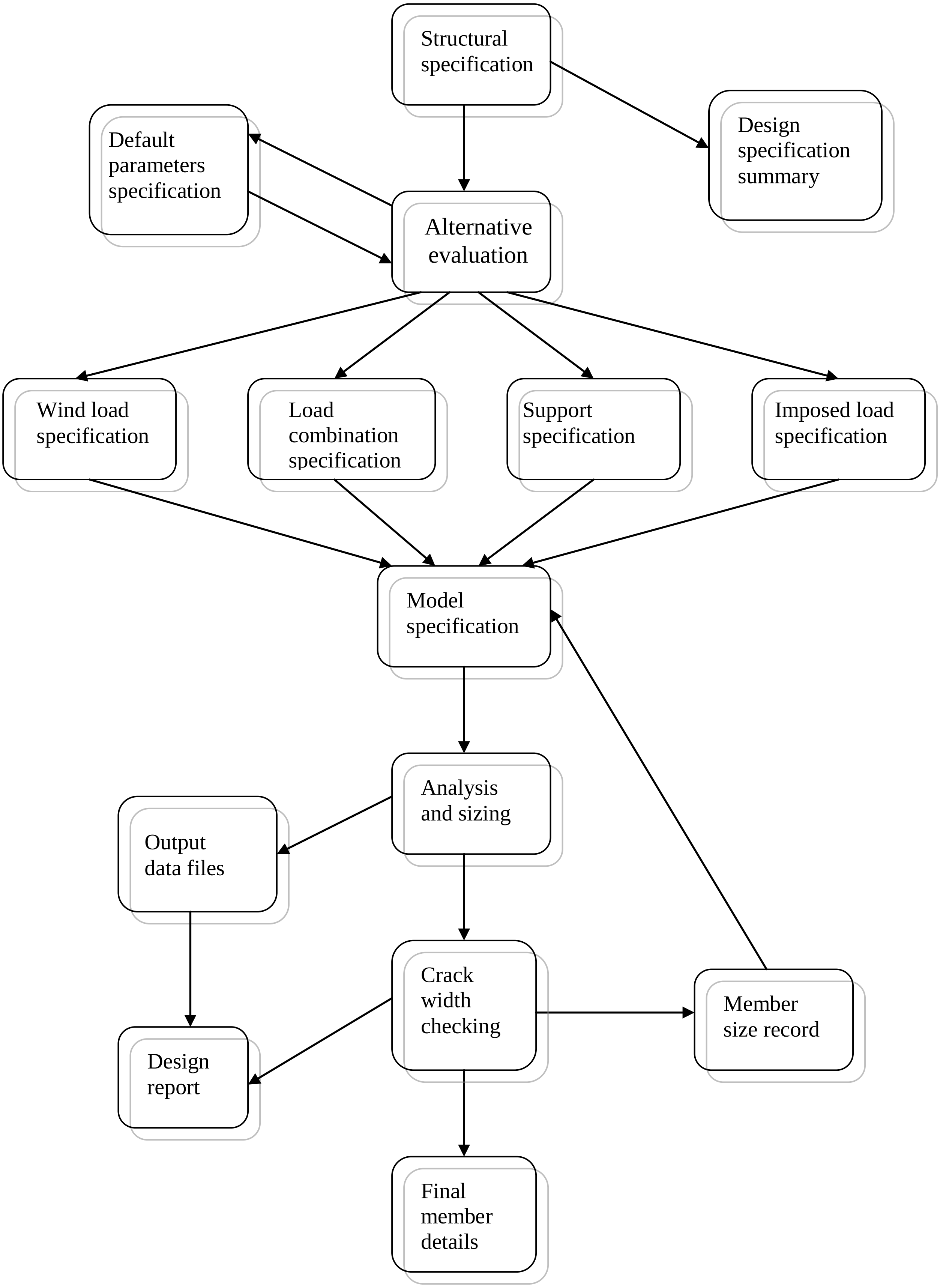

Figure 3. Representation of key design process by semantic network

```
Private Sub OptCircular_Click()
   CmdSearchForConfigurations.Enabled = False
   BBConfigurationRequirement.Shape = "circular"
   ImageShape.Picture = LoadPicture("\mydocu~1\kbs\Circular.bmp")
   Unable2TextLabels TxtWidthBreadthRatio, LblWidthBreadthRatio
   LblDiameter1.Visible = True
   LblDiameter2.Visible = True
   LblDiameter3.Visible = True
   If TxtHeight.Text <> "" And TxtVolume.Text <> "" And _
      TxtDensityOfLiquid.Text <> "" Then
      LblDiameter2.Caption = _
         Format$(BBConfigurationRequirement.Diameter, "##0.##")
      CmdSearchForConfigurations.Enabled = True
      CmdSearchForConfigurations.SetFocus
   End If
   UncolorAllButtons
End Sub
```

Figure 4. A typical example of procedural method attached to optional button OptCircular in preliminary design screen

```
!RULE GROUP: height aspect factors Ch OF BB Wind Load

RULE to find height aspect factor Ch : 1 of 6
IF heightToBreadthRatio OF BBLiquidRetainingStructure <= 1.0
THEN heightAspectFactorChx OF BBWindLoad := 0.95

RULE to find height aspect factor Ch : 2 of 6
IF heightToBreadthRatio OF BBLiquidRetainingStructure > 1.0
AND heightToBreadthRatio OF BBLiquidRetainingStructure <= 2.0
THEN heightAspectFactorChx OF BBWindLoad := 0.05 * heightToBreadthRatio OF
BBLiquidRetainingStructure + 0.9

RULE to find height aspect factor Ch : 3 of 6
IF heightToBreadthRatio OF BBLiquidRetainingStructure > 2.0
THEN heightAspectFactorChx OF BBWindLoad := 0.025 * heightToBreadthRatio OF
BBLiquidRetainingStructure + 0.95

RULE to find height aspect factor Ch : 4 of 6
IF heightToLengthRatio OF BBLiquidRetainingStructure <= 1.0
THEN heightAspectFactorChy OF BBWindLoad := 0.95

RULE to find height aspect factor Ch : 5 of 6
IF heightToLengthRatio OF BBLiquidRetainingStructure > 1.0
AND heightToLengthRatio OF BBLiquidRetainingStructure <= 2.0
THEN heightAspectFactorChy OF BBWindLoad := 0.05 * heightToLengthRatio OF
BBLiquidRetainingStructure + 0.9

RULE to find height aspect factor Ch : 6 of 6
IF heightToLengthRatio OF BBLiquidRetainingStructure > 2.0
THEN heightAspectFactorChy OF BBWindLoad := 0.025 * heightToLengthRatio OF
BBLiquidRetainingStructure + 0.95
```

Figure 5. A typical example of rule set for determination of height aspect factors in wind load specification

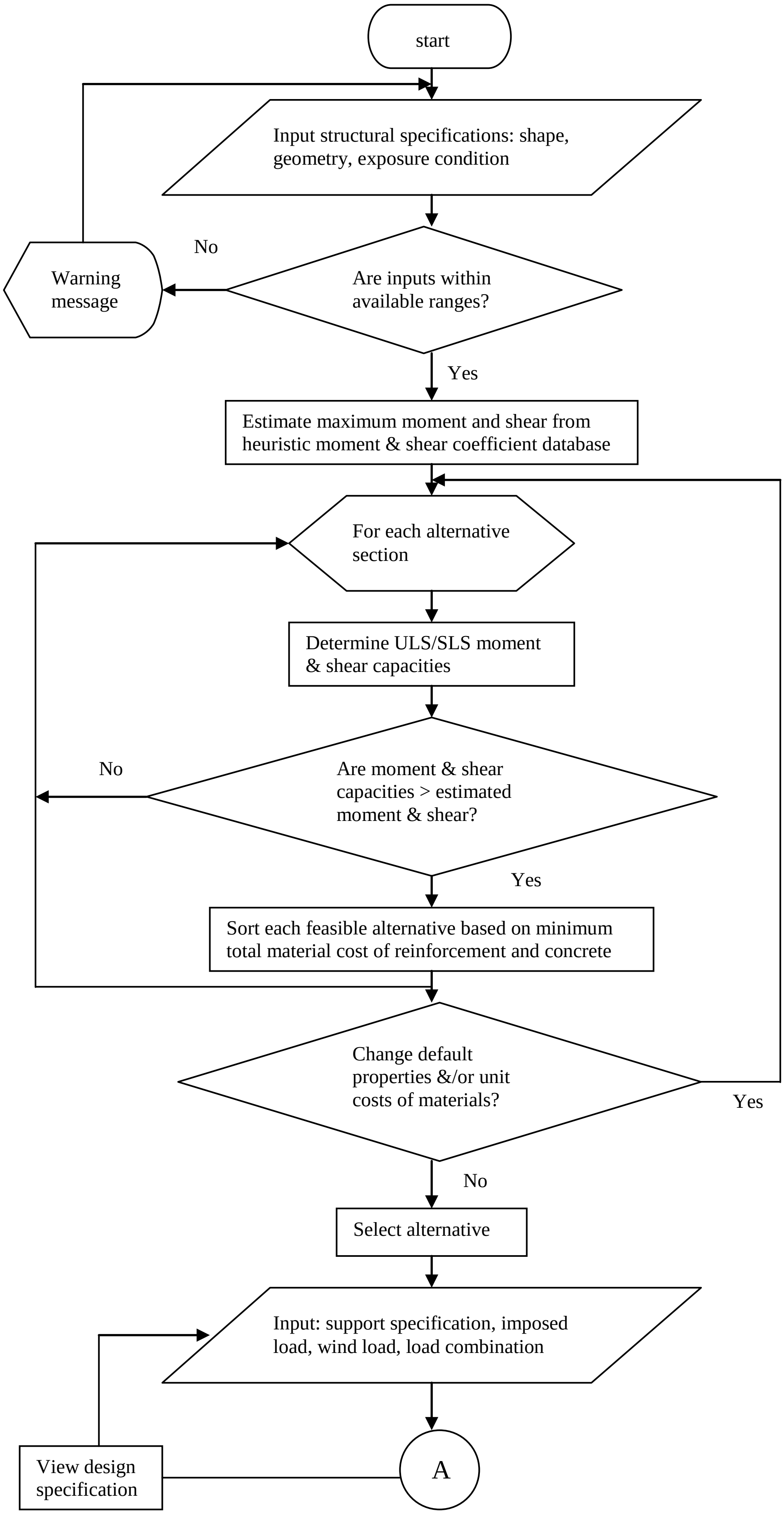

Figure 6(a). Flowchart showing overall design algorithm of prototype system

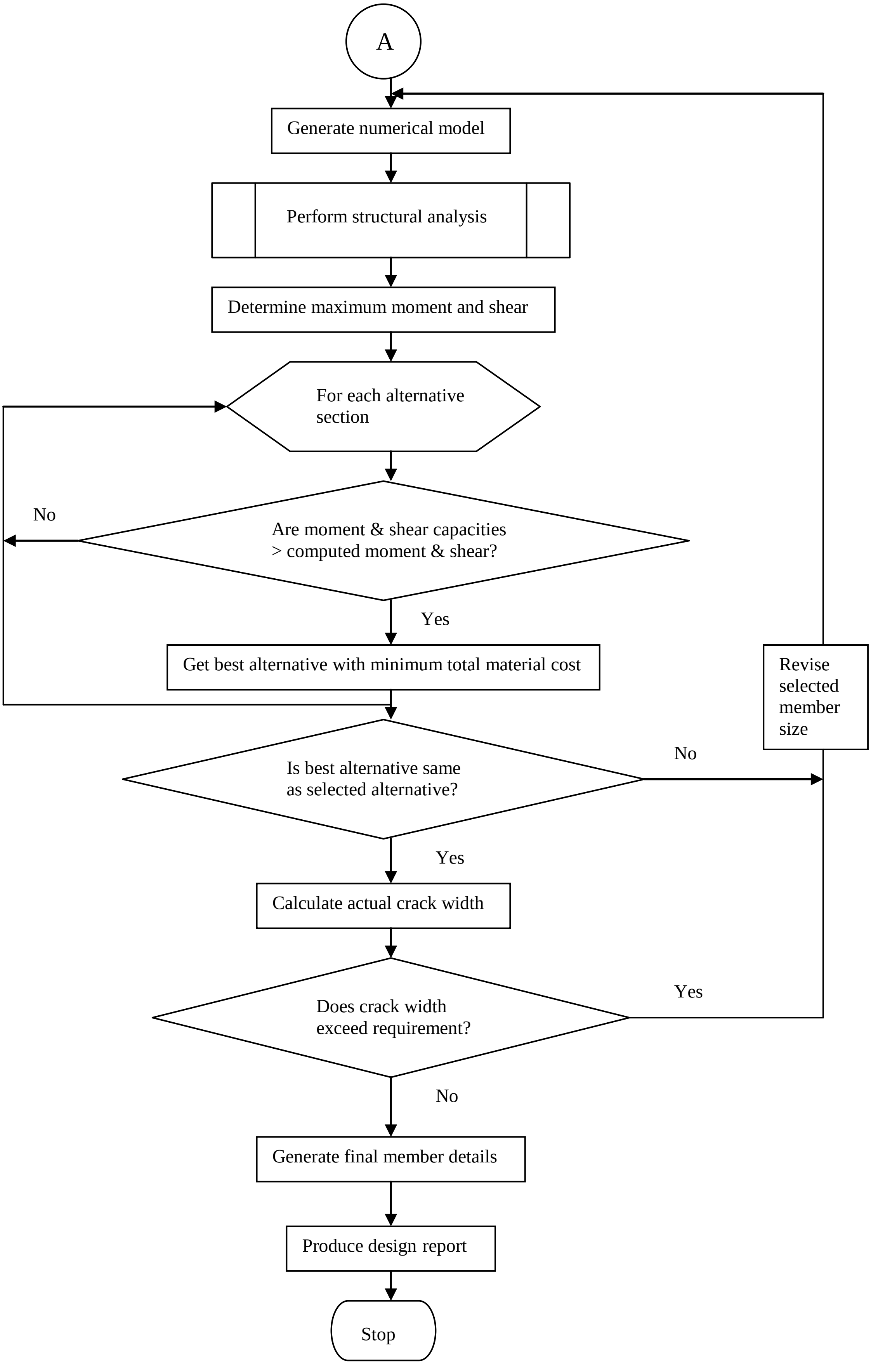

Figure 6(b). Flowchart showing overall design algorithm of prototype system

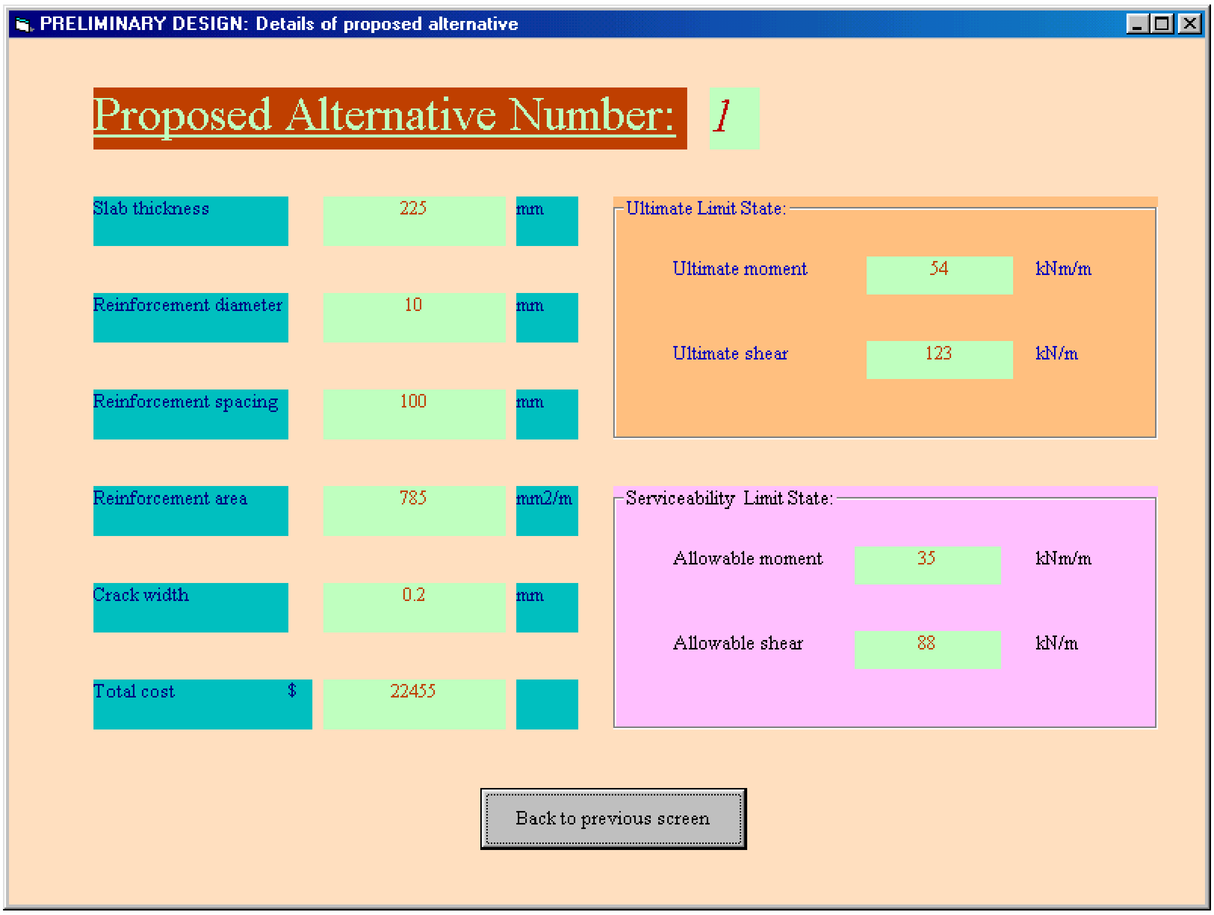

Figure 7. Screen showing details of proposed alternative

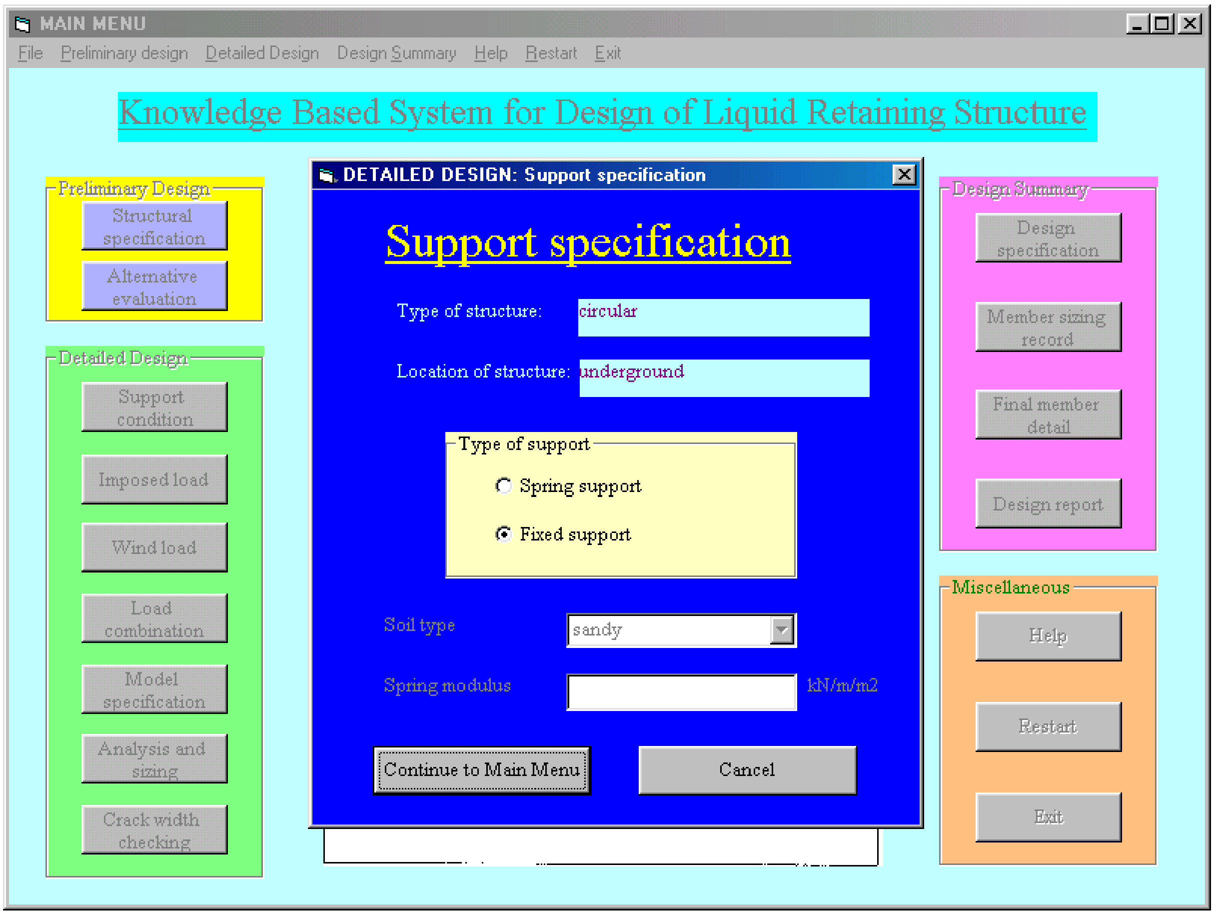

Figure 8. Screen displaying support specification