A knowledge-based system for design concept negotiation

Y. Cai and D. Martinelli
Department of Civil Engineering
West Virginia University
Morgantown WV 26506
USA

1. Introduction

Conceptual Design is a preliminary stage of system design. It involves feasibility study, system specification, decision making and primary design. As system design projects become large, a multidisciplinary approach is necessary to accomplish the task. An important activity of the cooperative conceptual design is concept negotiation. In this recursive process, design members evaluate and integrate the proposed design concepts from different perspectives until the design objective and constraints are satisfied.

The traditional way for the concept negotiation is the so-called "brain storm", which typically takes place at regular project meetings. For example, designers present their concepts with transparencies, blackboards, and drafts in the meetings, while other group members review the concepts, and then some members generate new concepts. As the number of the generated concepts increases, the inherent weakness of the traditional brain-storming technique presents its disadvantages: First, there is little systematic guidance for the concept generation. For example, some designers neglect the concepts that they are not quite familiar with. Second, there are few systematic methods for concept evaluation. This is because designers from different disciplines use different evaluating value systems, measurements, and terms. Third, it is difficult to trace the design concepts and to know "Where we are?". For example, a design concept generated during previous meetings may be "forgotten" for a period, and then be proposed again as a new concept. Also, it is difficult for the designers from one team to know what is going on in other teams.

---

[1]Yang Cai, Research Assistant, West Virginia University, Department of Civil Engineering, Morgantown, WV 26506, 304-293-3031

[2]David Martinelli, Assistant Professor, West Virginia University, Department of Civil Engineering, Morgantown, WV 26506, 304-293-3031

In this paper, knowledge-based system tools are studied as a means to improve the efficiency of the design concept negotiation. It consists of an object-oriented knowledge base, decision supported models, and user interfaces. The function of the system is to provide a support for representing design concepts, reviewing design concepts, and decision making. A practical case study, "Remote Mining System for In-Situ Waste Containment," is presented as an illustration of the methodology.

2. Features of Concept Negotiation

A design concept can be defined as a design idea or an alternative for solving a problem. Design concept negotiation is a group discussion activity that involves design concept generation, evaluation and integration from different prospects of design teams. Normally, the design members review their current design concepts based on the system objectives and constraints. They make decisions for (1) selecting the concept, (2) integrating the concept with other concepts into a new concept, or (3) rejecting the concept. Figure 1 illustrated the process of a concept negotiation. This "evolutionary" process is repeated through the entire project period.

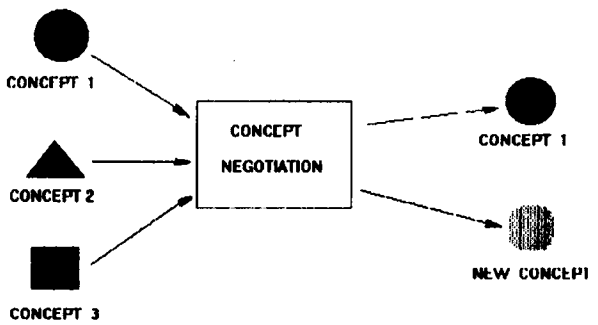

Figure 1. Concept Negotiation

Figure 2 shows an example of the dynamic pattern of concept accumulations before and after concept negotiations. It is found that, first, most stages of concept negotiation is a somewhat routine process; Second, a record system is necessary to handle the changing design concepts as the design concept progresses; Third, some simple decision making tools are useful for selecting concepts, such as priority ranking methods and decision tree etc. These tools are heuristic in nature, and represent potential for the application of a knowledge-based system.

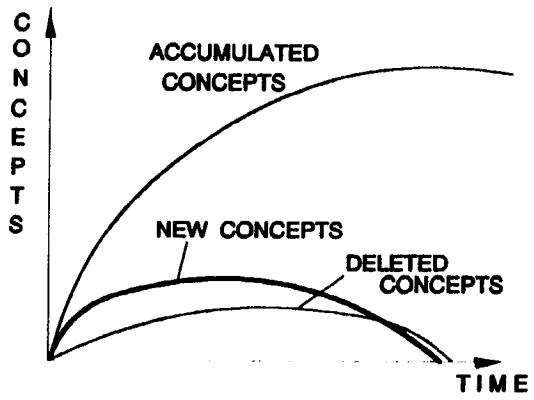

Figure 2. Concept Accumulation

3. The Knowledge-Based System

The knowledge-based system (KGS) is explored for improving the efficiency of cooperative design. The functions of the system are (1) to record, organize, and present design concepts; (2) to assist the day-to-day process of concept negotiation; and (3) to provide models, such as Weighted Average, Decision Tree, for decision making.

3.1. Scheme of the System

The system consists of four parts: (1) An Object-Oriented Database, which stores design concepts as objects or instances; (2) A rule base, which contains of high-level of heuristic knowledge for questioning, imputing, retrieving, presenting design concepts, also the rules for applying the decision model; (3) A decision models, and (4) A user interface.

3.1.1 Object-Oriented Database

The knowledge base component of the system is CLIPS 5.1. It is a forward-backward chaining rule-based language that has Object-Oriented Database (OOD) feature. The OOD uses ***Objects*** to store not only information, but also relationships between objects. In the system, long-term design concepts, such as major branches of total systems, are stored as objects. The relevant attributes or criteria are stored in ***Slots***. Classes of objects are arranged in a hierarchy or in a graph to describe the relationships of objects in a system. One of features of object-oriented databases, ***Inheritance***, is incorporated in this system. A subclass inherits attributes from one or more superclasses. Version 5.1 of CLIPS support only ***Is-a*** links. The ***Is-a*** link indicates the inheritance of slots from a class to its subclass. The inheritance of slots is particularly important in the system since it means that we do not have to redefine the properties and behavior of each new class of objects that are defined. Also it means all inherited criteria should be considered by designers at any subclass level. The format of an object definition:

```
(defclass <object name>
  (is-a <superclass>
  (slot <attribute 1>        [optional]
       <default value>       [optional]
       ...
  (slot <attribute n>
       <default value>)))    [optional]
```

According to CLIPS, an ***Instance*** is an object that has values. In this system, an instance is used to store temporary design concepts as well as other knowledge. The format of an instance definition is:

```
(definstances <instance-group-name>
(<instance-name> of <object-name>
                (<slot-name> value)  [optional]
          ...
                (<slot-name> value))  [optional]
        ...       ...
(<instance-name of <object-name>
                (<slot-name value)    [optional]
                  ...
                (<slot-name> value))) [optional]
```

3.1.2 Rule Base

The rule base contains the rules about concept retrieval, design rationale, questioning, concept evaluation etc. The format of a rule is as following:

```
(defrule <rule-name>
   (<patterns>      [left-hand side of the rule; if...]
   =>
   (<actions>))     [right-hand side of the rule; then...]
```

3.1.3 Decision Models

In this system, types of decision models include: Weighted Averages, Morphologic Models, Decision Trees and Analysis of Hierarchical Process (AHP). Morphological Model is used to generate all possible combinations of alternatives. It is straightforward with the *do-for-all-instances* function in CLIPS. Weighted Average models are used for selecting alternatives based on designers opinions. It is also accomplished with CLIPS. Also, rules are used to transfer verbal descriptions into numbers. The Decision Tree is an external C routine that can be used to select alternatives based on given probabilities and numerical values. Also AHP is a C routine that is used for selecting alternatives by comparing them sequentially.

3.2 Communication Modes

Interacting the system with the design teams is important because the system cannot work without human-computer interaction. The system operates using three communication modes: First, in single mode, a project coordinator runs the system and prints out results, and then presents them when the project teams meet; Second, in presentation mode, the system presents the-state-of-the-art during project meetings through a computer screen projector; and third, in network mode, users may interact with the decision support system through a network any time. See Figure 3.

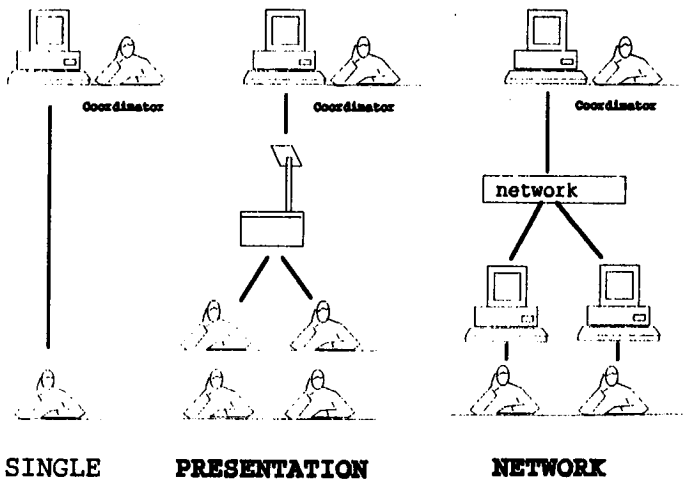

Figure 3. Application Modes

4. Application

The application of this study is based on an ongoing multidisciplinary project: "Remote Mining for In-Situ Waste Containment", at West Virginia University. This project focuses on is installation of wide-area, three-dimensional containment systems. The current system design concepts are shown as Figure 4. These are expected to extend up to several hundred meters below the ground surface and to be emplaced robotically to minimize human exposure to hazards. These containment systems could be implemented as stand-along barriers to control toxic plume migration. The project is divided into three subtasks: mining systems, waste containment technology, and automation systems. Designers of this project come from many disciplines: Civil Engineering, Mining Engineering, Electrical and Computer Engineering, Environment Engineering, also Mechanical Engineering. At the current conceptual design stage, the

weekly project meeting is the basic media for design concept negotiation. The dynamic pattern of the accumulated design concepts and an illustration of current system design are shown as Figure 4 and 5.

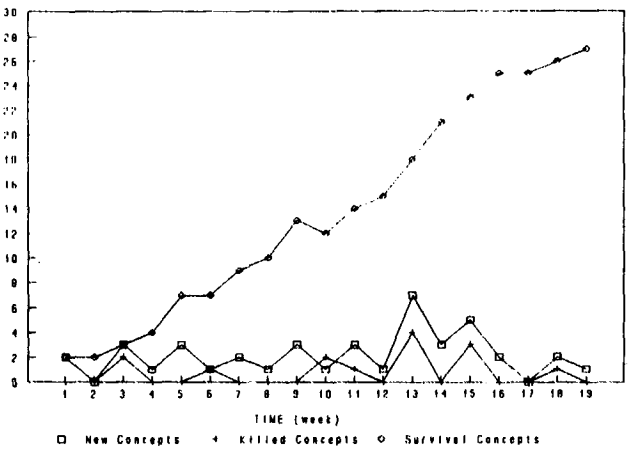

Figure 4. Concept Accumulation

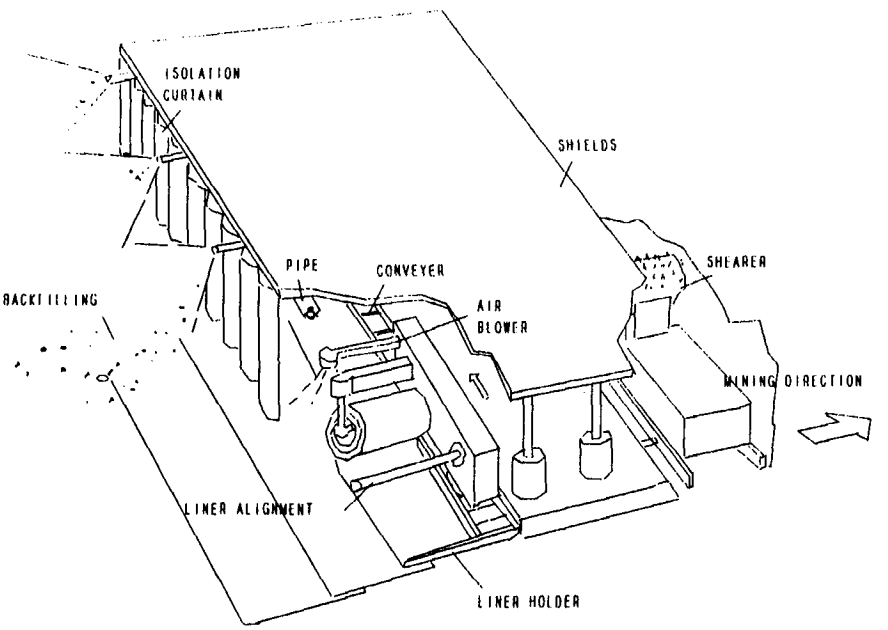

Figure 5. Illustration of the System

The Object-Oriented Knowledge Base is used for representing the knowledge of design concepts, the knowledge of questioning, also the knowledge of decision making. Figure 6 shows a simplified structure of the major design concepts. Samples of CLIPS codes for the knowledge representation and output are presented as followings.

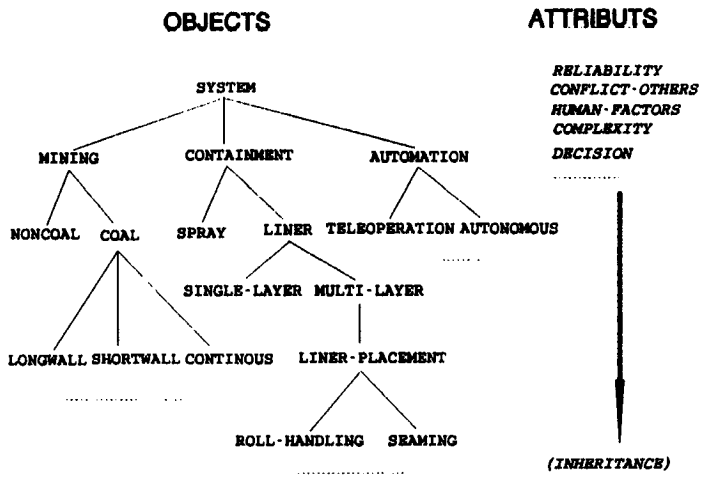

Figure 6. Objects of Design Concepts

```
(defclass SYSTEM (is-a USER)
    (abstract)
     (slot conflict)
     (slot reliability)
     (slot human-factors)
     (slot availability)
     (slot decision)
        (default pass)))
  (defclass MINING (is-a SYSTEM))
  (defclass CONTAIN (is-a SYSTEM))
  (defclass AUTOMATION (is-a SYSTEM))
  (defclass NONCOAL (is-a MINING))
  (defclass COALMINING (is-a MINING))
     (defclass LONGWALL (is-a COALMINING))
     (defclass CONTINUOUS (is-a COALMINING))
  (defclass SHORTWALL (is-a COALMINING))
```

```
(defclass SPRAY (is-a CONTAIN))
(defclass LINER (is-a CONTAIN))
  (defclass SINGLE-LAYER (is-a LINER))
  (defclass MULTI-LAYER (is-a LINER))
    (defclass LINE-DIRECTION (is-a MULTI-LAYER))


(defclass HUMAN_MACHINE (is-a AUTOMATION))
(defclass AUTONOMOUS (is-a AUTOMATION))
```

A sample of input instance of liner placement alternative:

```
(definstances CONTAIN-OBJECTS
  (PARALLEL of LINE-DIRECTION
                (conflict low)
                (reliability high)
                (human-factors mid)
                (availability high))

  (PERPENDICULAR of LINE-DIRECTION
                (conflict high)
                (reliability low)
                (human-factors high)
                (availability high)))
```

The following is the output of the structure of the concepts:

```
SYSTEM
  MINING
    NONCOAL
    COALMING
      LONGWALL
      CONTINUES
      SHORTWALL
  CONTAIN
    SPRAY
    LINER
      SINGLE-LAYER
      MULTI-LAYER
        LINE-DIRECTION
  AUTOMATION
    HUMAN-MACHINE
    AUTONOMOUS
```

In addition, the structure of questioning or reviewing:

ASK
  GENERAL
    WHAT
    WHY
    WHERE
    WHEN
    HOW
  CONSTRAINT
    AVAILABILITY
    TIME
    COST
    LABOR
    RELIABILITY
  OBJECTIVE
    FAST
    ECONOMY
    RELIABLE
    LESS-LABOR
    COMPLEXITY
  SPECIAL
    MINING
    AUTOMATION
    CONTAINMENT

Applying the query system of Clips 5.1, six major query functions are available: 1) *any-instancep*, determines if one or more instance-sets satisfy a query; 2) *find-instance*, returns the first instance-set that satisfies a query; 3) *find-all-instances*, groups and returns all instance-sets which satisfy a query; 4) *do-for-instance*, performs an action for the first instance-set which satisfies a query; 5) *do-for-all-instances*, performs an action for every instance-set which satisfies a query as they are found; and 6) *delayed-do-for-all-instance*, groups all instance-sets which satisfy a query and then iterates an action over this group.

5. Conclusion

In this paper, a prototype of a knowledge-based system for supporting design concept negotiations is studied. The object-oriented database, rule base, and decision model base form an intelligent knowledge base so that the system can improve the design concept communication and decision making. However, this system is in its prelimary stage. Further study about the human-computer interaction as well as graphical interface are necessary in order to accurately assess the potentials of such a system.

References:

Giarratano, J.C., CLIPS User's Guide, COSMIC, 1991

Klein, M.(1993), Capturing Design Rationale in Concurrent Engineering Teams, *Computer*, **January, 1993**

Londono, F.(1990), A Blackboard Framework to Support Concurrent Engineering, Dissertation, West Virginia University

Londono,F. (1989), etc., A Blackboard Scheme for Cooperative Problem-Solving by Human Experts, *Proceedings of the MIT-JSME Workshop on Cooperative Product Development*, **1989**

Reddy,Y.V., Srinivas,K., etc.(1993). Computer Support for Concurrent Engineering, *Computer*, **January, 1993**

Sobolewski, M. (1991), Object-Oriented Knowledge Bases in Engineering Applications., *Proceedings of 6th International Conference on CAD/CAM Robotics and Factories of the Future*, London, UK, Aug. 19-22, 1991

Sobolewski, M. (1992), Graphical User Interface for Collaborative Work, CERC Technical Report, Concurrent Engineering Research Center, West Virginia University, Morgantown, WV

Tong, C. (1993), A Knowledge-Based Computer Environment for the Conceptual Design of Small Electromechanical Appliances, *Computer*, **January, 1993**