Knowledge-Based Analysis of Structural Systems

by

G. M. Turkiyyah and S. J. Fenves

EDRC-12-12-87

Knowledge-Based Analysis of Structural Systems

George M. Turkiyyah and Steven J. Fenves

Department of Civil Engineering
Carnegie Mellon University
Pittsburgh, PA 15213-3890
USA

Abstract

The effective use of finite element systems relies on the availability of physical modeling and result interpretation capabilities. However, reliable computer tools that assist in the modeling and interpretation tasks are still inexistant. In this paper, we investigate some issues underlying the development of an integrated knowledge-based structural analysis assistant and propose an architecture, based on the control blackboard model, for organizing the diverse sources of knowledge that are needed in such an environment.

Table of Contents

List of Figures

Knowledge-Based Analysis of Structural Systems[1]

George M. Turkiyyah[2] and Steven J. Fenves[3]

Department of Civil Engineering
Carnegie Mellon University
Pittsburgh, PA 15213-3890
USA

1. Introduction

The goal of structural analysis is to understand the physical response of structural systems when subjected to loads. Modern structural analysis heavily relies on numeric models based on logical and rational theories that have been tested experimentally for their validity. In this context, the translation of a physical situation (structure and loads) into corresponding mathematical models and the mapping of the numeric results back to the physical response take on a primary importance. Computer tools are needed to assist in these tasks. Developments in AI techniques for representing and reasoning about physical systems make it realistic to try to partially automate these activities. In this introductory section we define the structural analysis problem, briefly discuss some previous systems that attempted to tackle the problem and present our motivations and objectives for undertaking this work.

1.1. The Problem

The classic definition of analysis refers to the task of solving a well-defined mathematical model. In this paper we use a more encompassing definition that globally decomposes the problem into three generic steps as depicted in Figure PROBLEM.

Modeling. This is the process of formulating a mathematical model that reflects the reality of the physical problem at hand. Modeling can be further subdivided into physical and numeric modeling. Physical modeling is the *idealization* of the problem by extracting from the physical structure the aspects of behavior that are deemed essential. The analytical model thus produced is a precise and unambiguous statement of the problem, including the relevant variables of the problem and the equations that relate them. Although a symbolic solution of the analytic equations might be possible in simple problems, usually the formulated problems are more general than can be directly solved, so that numeric models have to be resorted to. Various numerical modeling techniques, differing in generality and ease of application for particular problems are available (finite strips, boundary elements, finite elements, etc.). In structural analysis, finite element models are by far the most widely used and we will limit ourselves to them.

---

[1] **Submitted to the Second International Conference on the Applications of Artificial Intelligence in Engineering, Boston, USA, August 1987.**

[2] **Research Assistant.**

[3] **Sun Company University Professor.**

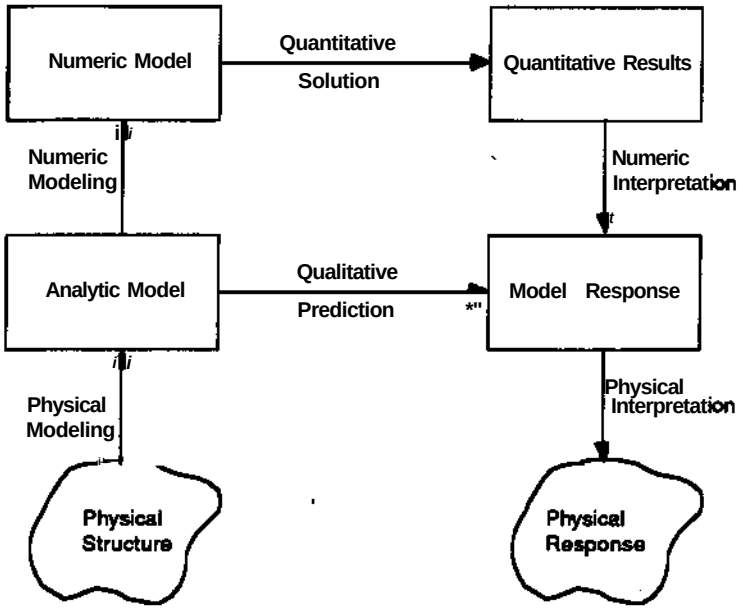

Figure 1-1: The Structural Analysis Problem

Solution. This is the process of solving the equations defined by the numeric model. It is a computationally expensive process but, in the current state of the art, has been largely automated. Many comprehensive commercial packages are available for the solution of a wide range of numeric models. However, some special problems might require human interaction, such as determining precision, adjusting increments, controlling convergence rates and choosing efficient algorithms.

Interpretation. The solution of the numeric model being just a set of numbers, there is a need to relate the numeric results back to the real problem that was idealized by the model. This process is closely related to the modeling step; in fact, it is roughly its dual, and is subdivided into two steps. The first step, model response abstraction, consists of the numerical interpretations which detect characteristics, trends and properties of the solution, match them with expectations about the model behavior and justify the assumptions built into the model. The second step, physical response abstraction, consists of the physical interpretations which relate the model quantities to the physical structure. A particular idealization implicitly defines a set of expectations that should be mapped back from the model solution to the original structure. In a broader context, interpretation gives directions for changing, improving or refining either the structural system being investigated or the model that was used for the analysis, or both.

1.2. Previous Approaches

The bulkiness of finite element packages, the difficulty in learning to use them, their potential misuse, and the need to interface them efficiently with design programs, coupled with the fact that aspects of the modeling and interpretation process have a heuristic nature, were among the motivations to apply knowledge-based techniques to build analysis assistants.

SACON [Bennett78], the earliest structural analysis consultant, was the first application of the EMYCIN environment to non-medical domains. SACON addresses the numerical modeling aspects of the problem, using the heuristic classification problem solving paradigm [Clancey85]. A simple real-valued-parameter model of the behavior of the structure is abstracted and important features of the structural behavior (controlling stress, deflection and non-linearities) inferred from it. A heuristic match of these features with various analysis options is then performed. FEASA [Taig86] is an analysis assistant that addresses the physical modeling aspects of the problem. FEASA incorporates a large number of heuristics that give recommendations about problem specification, high level modeling, mesh size, etc. through a question/answer session. The system is limited by the knowledge representation and structuring capabilities of the SAVOIR shell in which it is implemented. A third analysis consultant is PLASHTRAN [Cagan87], a modeling aid for plates and shells for analysis using the MSC/NASTRAN code. PLASHTRAN recommends analysis strategies and program options by mapping problem features to the recommendations available in its knowledge base. The features are asked from the user via a series of tailored menus.

Others researchers have investigated the possibility of integrating finite element programs with modeling and interpretation modules in a design environment. Among those efforts are FACS and CARTER. FACS [Gregory86] is a system under-development to convert airplane designs to finite element models and solve the models produced. FACS takes its input from a database that contains information about the design. From this data (typically expressed in terms of points, lines and surfaces), the system constructs a higher level description of the structure and its various components. The components are classified according to their types and the functional roles they play in the design and then matched with corresponding analysis methods and parameters in the knowledge base. Special-purpose algorithms are then exercised to produce the finite element meshes. CARTER [Reynier86] is an expert system for the analysis and dimensioning of casings of mechanical components such as transmission engines. CARTER uses two kinds of analysis models: a strength of materials model to perform predimensioning in a preliminary stage and a finite element model to check the dynamic behavior in a detailed stage. Although CARTER has no substantial capabilities to refine the finite element model when it is unsatisfactory, it does have capabilities to use the finite element analysis results to guide the design process.

Although the systems reviewed above have achieved some success in their application domains, they leave a lot to be desired from a structural analysis assistant.

First, the systems presented can be cast into the classification problem solving paradigm [Clancey85]. Models are not constructed but selected. The systems effectively act as big switches for models and algorithms and are limited to narrow application domains.' The systems use productions to fill the parameters of complete model templates based on a set of a-priori chosen features. They do not explicitly reason about the functionality of the structural systems they analyze and how the functionality is achieved by the interaction of components that individually behave in some fashion.

Second, the knowledge incorporated in these systems is mostly empirical in nature, representing the experiential knowledge of their developers. This makes it difficult to generalize the knowledge and leads to an explosion of the knowledge base size as more cases are covered. Although, admittedly, there is an empirical component in modeling, there are other kinds of reasoning on top of which heuristics can be built and that can make the expression of those heuristics more compact. In particular, the exclusive use of empirical knowledge leads to a system that can presumably solve difficult problems but stops short of reasoning about much simpler problems. Furthermore, the nature of the empirical knowledge is typically specific and inflexible, and does not reflect the fact that many chunks of the analysis knowledge are generic and are used by humans for various purposes including checking, evaluation and design.

Third, the systems presented do not treat assumptions explicitly, even though assumptions about the behavior of the structural system are the basic entities from which models are constructed, evaluated and judged adequate. The lack of explicit treatment of assumptions is mainly due to the absence of a formalism for expressing the assumptions in terms of behavior modes and abstract concepts. Such an omission obscures the knowledge incorporated in these systems and raises questions about their reliability. This omission also leads to an inability to hierarchically refine the analysis models, because a single and complete model that implicitly makes many assumptions about the structure is instantiated. Thus no reasoning can be performed about how individual assumptions affect the form of the model and the eventual analysis results.

Fourth, the interpretation capabilities of these systems are limited. In particular, the interpretation is either treated as a unilateral post-process or relegated to the design program that uses the results. The capability to evaluate the chosen analysis, strategy and integrate the interpretation of the model solution into another pass that refines the model, is not addressed. In fact, the problems mentioned in the previous paragraphs compound, and make it very difficult to incorporate such feedback.

1.3. Motivation and Objectives

Because the current approach to building knowledge-based analysis assistants results in systems that tend to be hand-crafted for specific structures and specific modes of interaction with the user, and that intricately mix diverse types of knowledge, we feel that there is the need for a deeper approach to the structural analysis modeling and interpretation tasks.

The approach we are pursuing uses formal structural principles of behavior models in the system's core and incorporates empirical knowledge in the form of assumptions that are built on top of this core. We intend to represent structural systems at three distinct levels: compositional (spatial representation), functional (conceptual organization) and behavioral (load-deflection response). The generation of numerical models is conceptually treated as a subordinate process driven by the need to quantify the charateristics of a behavior model generated from the physical structure through the application of behavioral assumptions. The modeling and interpretation tasks are not treated independently: they access the same information and the scope of the interpretations is defined during the generation of the model.

By treating the analysis problem with a formative problem solving approach, whereby models are recursively *constructed* from lower level components under various behavioral assumptions, more flexibility in model generation capabilities can be achieved than with the classification problem solving approach.

2. Elements of an Approach

In this section we examine some issues that confront the development of general purpose structural analysis assistants. We present the principles and ideas that underlie the development of our system and discuss the advantages they provide.

2.1. Representing Geometry

The spatial representation of structural systems is of primary concern, for two reasons. First, since the objects dealt with in structural analysis are physical (in the sense that they all have some spatial manifestation), it is natural to consider their spatial representation (geometry and topology) as a basic descriptive vehicle. From this point of view, every structural system is composed of a set of distinct components connected in some fashion, and every component is specified by a unique geometric description representing the volume that it occupies. Second, geometry is the level at which an analysis assistant can interact with a finite element program for the numeric solution.

For structural analysis, we draw a distinction between object geometry, model geometry and finite element mesh geometry. Although they might, in some cases, be syntactically equivalent, the semantic content of the information associated with them is quite different. The object description is the "natural" representation of the system and essentially reflects how the information about the structural system would be communicated to, say, the manufacturing or construction process. The model geometry is part of the physical idealization of the system and is *derived from* the object geometry by various idealization schemes. The model dimensionality is often different from the object dimensionality. The finite element mesh geometry is, in general, an approximation of the model geometry. Direct generation of finite element meshes from object models, as available in CAD systems often poses difficulties, because of the "missing level" that the model geometry represents. We want to explicitly represent the fact that the finite element solution approximates an already idealized model and not the physical object directly. Only in very detailed models, where the object and the model have similar spatial representations, can a direct CAD-FEM translation be meaningful.

Although geometric modeling and geometric reasoning are an intrinsic part of an analysis system, it is not the purpose of this paper to discuss these issues. We mention, however, a few points about the object geometric descriptions that affect the rest of the system. First, the geometric specification language should allow the combination of well-defined primitives into more complex objects, not just the instantiation of a complex geometric model. Second, the primitives should be meaningful, i.e., representing parts that are easy to describe, understand and reason about, from a user point of view. Third, the ability to describe a structure at various levels of geometric abstraction and recursively

subdivide it, is highly desirable because it allows the early identification of features that are often useful clues for modeling. Finally, a very important characteristic of the geometric representation is that it should be independent of any physical idealization and should make no assumptions about contextual or functional interpretations. These later enrichments of the descriptions represent added knowledge that should be separately and explicitly represented. Such independence guarantees that all idealizations could *potentially* be produced. This completeness criterion is highly desirable in a system that aspires to be easily expandable.

2.2. Functional Descriptions

The spatial description of a structural system provides only a weak representation of the system. The spatial representation is flat and does not provide insights into the underlying organization of the structural system. Engineered systems are designed to fulfill specific functions; substantial knowledge about the roles of the various parts and their intended purpose is implicit in this organization. This information corresponds to a different kind of knowledge that we propose' to encode in a separate representation: the functional description.

In structural analysis, the functional description reflects how the structure is conceptualized to resist applied loads. The functional description of a structural system is represented along a horizontal dimension and a vertical dimension. The horizontal dimension corresponds to different views of the same structure (or substructure) as derived from the various functional roles it plays, i.e., how it resists various loads. We use the term functional models to refer to these different views. The vertical dimension corresponds to hierarchical abstractions of the functional models of the structural system.

Functional models are needed for pragmatic reasons only because, in principle, knowledge of the geometric and material properties of a structure is sufficient to analyze it. However, the representation and use of functional models offer many advantages. First, functional models reduce the complexity in analyzing structural systems, because these models impose a hierarchical conceptual organization on the geometric model. Second, functional models allow a more meaningful user interaction, because the primitives of the functional description are the entities that engineers commonly use to describe structural systems. Third, functional models are pointers to a large body of heuristic knowledge about the behavior of a structure, because knowledge of the purpose of a component has a direct impact on the way that component should behave to achieve its purpose. Fourth, functional models provide the link to synthesis and design programs, because in the early stages of design information about functionality is available before the structure is specified, and it is the analysis results that guide the design process.

2.3. Assumptions underlying models

All structural analyses make assumptions. In fact, it could be argued that structural mechanics knowledge is either knowledge of assumptions or expertise in mathematics (symbolic algebra and theorem proving). Traditionally, however, modeling has been done by selecting between a fixed set of models, each of which is applicable to some set of geometric and functional features and includes a "packaged" set of

assumptions built into it, often not explicitly stated. Such a mode of operation usually result either in an oversimplification of the problem or in needlessly expensive models. The explicit manipulation of behavioral assumptions not only allows the identification of the premises on which the analysis is based, but also provides fine-grained control over the models produced [Wellman86], by tuning individual features and aspects of the model according to the specifics of the problem at hand.

In structural analysis, assumptions refer to behavioral constraints on the components and subsystems, and represent a variety of insights about the structural system's behavior (e.g., assumptions such as "core stiffness negligible", "uniform load distribution", "no out-of plane deformation"). The correct interpretation of a particular list of assumptions gives rise to a model. Assumptions are the basic entities that determine the prediction capabilities and the limitations of a model. Furthermore, they can be expressed at a sufficiently abstract level to provide a compact and comprehendable set of criteria to evaluate a model. Assumptions are not derived from other principles, per se. They are expectations projected to hold, and are used either to limit the complexity of the model or because information about the design is incomplete. Thus assumptions play a dual role in analysis and design tasks. The geometric and functional organizations are the sources of the expectations and provide the justifications for their application. Assumptions differ in their reliability; depending on the problem they might or might not prove adequate by later elaboration of the design or refinement of the model.

To be able to add and retract behavioral assumptions dynamically during solution, we associate a confidence value and an evaluation criterion with each assumption. Confidence values reflect the a-priori credibility rating that a particular expert might assign to various assumptions. These ratings are likely to be different between different modelers. The evaluation criteria, on the other hand, are expressions that can be evaluated a-posteriori (after the numerical model is solved) to check the validity of the assumptions. Some of the modeling assumptions can be checked by directly evaluating a predicate expression on the numeric results while others might require a separate analysis (presumably, smaller in scope) to be performed. Some assumptions, however, can not be validated formally except by going to a more refined model.

Modeling has often been compared to an art, yet acknowledged to rest on formal principles. We think that an explicit treatment of assumptions can bridge the gap between heuristic knowledge and structural theory, because behavioral assumptions specify the essential properties of the structure and relate them to the processes that produce models. Without this explicitness it is hard to define the capabilities of a modeling system because it is difficult to tell whether the models produced come about from individual heuristics directly built in the system or whether the models were derived by applying sound principles of more general applicability to the specific situation.

2.4. Behavior Models

The application of assumptions results in a *behavior model.* The behavior model is the link between the structural system as a physical object and the finite element model as a numeric model. The behavior model represents the abstract mathematical problem that the finite element model approximates. The primitives of the behavior model are derived from basic concepts of structural theory. These concepts represent the kernel of the system and are used to express structural problems, independent of partiular application domains.

Our representation of the behavior model is a constraint network. The constraints are the relations between the generalized degrees of freedom of the behavior model representing the load-deflection relations of components and substructures and the equilibrium and compatibility relations that should hold between these components and substructures. Operations on the constraint network (e.g., adding or suspendinf constraints, establishing or breaking links, assigning values to parameters) reflect the physical modeling decisions during the solution of a problem.

Behavior models are the means to store the evolutionary and hierarchical developments that occur during a sequence of the analyses of a physical structure. The evolution of a model can occur either by refining the load-deflection relations of the model components or by expanding the dimensionality (number of degrees of freedom) of the model components. The hierarchy of behavior models is represented by evolution constraints that map the degrees of freedom of a substructure to the degrees of freedom of its components. The evolution constraints serve two purposes. First, they can be used for the aggregation of components into substructures to produce more abstract behavior models and check the correctness of a refined analysis. Second, they can be used for the assignment of component quantities from substructures quantities. Clearly, for this assignment to be deterministic, additional information, design decisions, is needed.

Behavior models provide the context in which the generation of the finite element model occurs and reasoning about the finite element analysis results is performed. The generation of a finite element model is guided by the need to quantify the behavior model. Mapping operators translate the behavior model in terms of which abstract behavior modes and quantities of interest are stated, to a finite element model. The duals of these operators extract the results of the finite element analysis and map them back to the quantities of the behavior model. This process is in sharp contrast to traditional finite element post-processors which are completely distinct interfaces custom-written for every class of models. Qualitative prediction of the characteristics of the finite element results can be performed if the constraints of the behavior model are instantiated with qualitative versions of the relations that the constraints represent, i.e., relations of the same form but using non-numeric coefficients. Further, the behavior model allows the solution of simple problems to be performed by simple constraint propagation without the need to call a finite element program.

3. A System Architecture

We are building a system that incorporates the above ideas. In this section we outline the overall architecture and the elements of the system. Figure ARCHITECTURE shows the architecture. It is partly based on the blackboard control model [HayesRoth85] where a set of domain knowledge sources post, under a control strategy, information on a global data structure: the domain blackboard. The control strategy is dynamically created by control knowledge sources that post on a separate blackboard: the control blackboard. A simple scheduling mechanism monitors both blackboards and manages their knowledge sources. Domain knowledge sources use resources during their operations. Resources are the means to specify static declarative information to the system as well as to integrate finite element programs in the architecture. Aspects of the architecture are adopted from Fenves [Fenves86].

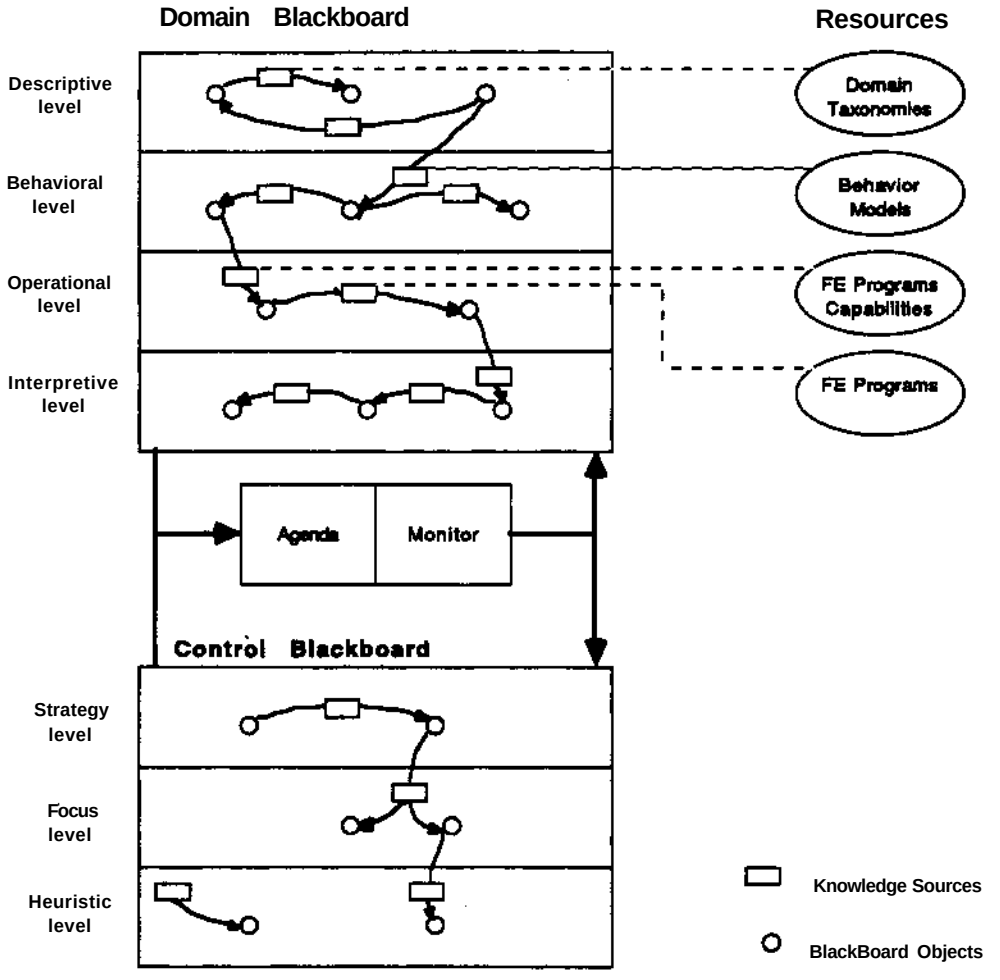

Figure 3-1: System Architecture

Syntactically, control knowledge sources are the same as domain knowledge sources; the only difference is that they operate on objects stored on different blackboards. While the decisions on the domain blackboard are operations on the structural system and its models, the decisions on the control blackboard refer to problem solving operations that make the application of particular domain knowledge

sources desirable. Domain knowledge sources are controlled indirectly because they respond to decisions made on the control blackboard. The architecture integrates domain and control problem-solving in a single basic control loop: at every cycle one knowledge source is chosen and applied, modifying a blackboard entry and triggering new knowledge sources to become candidates for execution at the next cycle. The knowledge source chosen at every cycle can either be a control or a domain source, i.e., control decisions compete with domain decisions as part of the solution process. Hence, the system can adapt its control plan to dynamic situations that occur during problem solving. Three domain independent control knowledge sources are responsible for scheduling: one updates the agenda at every cycle, another chooses a knowledge source to execute, and a third completes the cycle by transfering control to the chosen knowledge source for execution.

3.1. The domain blackboard

One way of visualizing the blackboard organization is along three dimensions. The "vertical*[1] dimension along four levels: descriptive, behavioral, operational, and interpretive, represents the object description, the behavior projection, the analysis model specification and the interpretation of the analysis, respectively. The "horizontal" dimension reflects the multiple views imposed on the structural system by the fact that it is designed to resist multiple loading conditions. This dimension represents the various functional decompositions, at the descriptive level; the different behavior modes that the structure exhibits in response to the various loads, at the behavioral level; the various numerical models in the analyses, at the operational level; and the different resulting interpretations, at the interpretive level. The "out-of-plane" dimension represents the elaborations and refinements that occur during problem solving. This dimension represents the hierarchy in specifying objects, at the descriptive level; the evolutionary behavioral abstractions and idealizations generated, at the behavioral level; the numerical refinements that occur on the analysis model, at the operational level; and the evolving interpretations, at the interpretive level.

Using this three-dimensional metaphor, we can talk about blackboard levels (the horizontal planes representing different kinds of information about the structure), blackboard slices (the vertical planes representing the different views of the structure) and blackboard elaborations (the normal planes representing hierarchical refinements). While the blackboard levels are fixed for structural analysis problems and the blackboard slices fixed for a given structural analysis domain (buildings, bridges, automobiles, etc.), the blackboard elaborations are not bound a-priori. It is the interpretation at every step that guides the succeeding elaboration that dynamically occurs until the interpretation is satisfactory. A description of the domain blackboard levels, the knowledge sources that post on them and the resources that the knowledge sources use during their operation follows.

The descriptive level holds the geometric and functional hierarchies of the structural system. This is the object level at which the information about the structure being analyzed is represented. The interaction between functional subsystems is represented by the fact that some of their components are linked to the same geometric description. The descriptive level is built by two knowledge sources that fit two usage

scenarios. First, in a top-down design context, a functional refinement KS develop the functional hierarchy "downwards" and link it to the geometric description. Second, in a design verification context, a geometry extractor KS derives suitable functional abstractions from a geometric description using a domain taxonomy resource that defines the various functional roles of structures and their subsystems.

The behavioral level holds models of the behavior of the structural system and its components, i.e., the generalized degrees of freedom and the load-deflection relations, as well as the relations between the hierarchy of models generated. Behavior models are built by knowledge sources that can generate, refine and abstract behavior models. These knowledge sources use a resource that contains the basic elements from which behavior models are built. Knowledge sources generate justifying behavioral assumptions for the actions they perform. The assumptions and the aspects of the behavior model that are entailed by the individual assumptions are maintained by an underlying assumption maintenance similar to ATMS [deKleer86].

The operational level holds a numeric model for input to a finite element program for quantitative analysis. The finite element model is a prescriptive sequence of operations that specify (potentially after some translation from a neutral format) the procedures to be executed by the target finite element program. These operations specify the finite element mesh, the material properties, the loading history as well as the response type, the solution algorithms and output options. Two knowledge sources build the finite element model: the first KS chooses the relevant options of the finite element program that are needed to reflect the characteristics of the behavior model, and the second KS generates the commands to the program.

The interpretive level stores three kinds of evaluations that answer three types of questions: does the numeric model correctly solve what the behavior model represents? does the behavior model reflect the correct physical behavior? does the structural system satisfy its intended function? It is the information at this level that determines whether backtracking or refinement are needed and if so, whether it is the numeric model, behavior model or object model that should be changed. Three knowledge sources perform the evaluation of the numeric results, model response and physical response, respectively. The evaluation of the numeric results involves issues such as accuracy and convergence of the analysis results. The evaluation of the analytical model involves the validation of the behavioral assumptions and of the confidence in the appropriateness of the behavior model with respect to the goal of the analysis. The evaluation of the physical response is done by comparing key response parameters to design specifications and functional criteria such as stress levels, ductility requirements and deflection limitations.

3.2. The control blackboard

The control blackboard is organized along three levels: strategy, focus and heuristic, representing different levels of abstraction of control decisions. The decisions at these levels determine the control plan, i.e., the desirable actions to be taken. Control knowledge sources that control model generation, monitor model evolution and strategize design modifications post their decisions on the control blackboard. A description of the control blackboard levels and the knowledge sources that post on them follows.

The strategy level records the global system-level tasks that are to be performed. These global problem solving decisions depend on the input of the problem, the goal of the analysis and how far in the solution process the system is. We define four kinds of strategies that: abstract a functional model and then generate a behavior model; build a geometric model and then a corresponding behavior model; refine a behavior model; and refine an object model.

The focus level records local goals, objectives and interests that focus attention to domain knowledge sources or blackboard objects during the solution to implement the active strategies. We currently recognize four KSs that implement the four strategies presented above. Focus decisions determine, for example, the order in which to consider the various subsystems, the granularity of the models and the nature of the analysis.

The heuristic level holds control heuristics that, strictly speaking, fall outside the scope of structural theory but are useful to obtain a solution in reasonable time and/or using limited memory requirements. The control knowledge sources that post at this level include miscellaneous heuristics that assign weights to various control decisions.

4. Conclusions

The analysis of structural systems is a task that combines quantitative and qualitative, symbolic and numeric, formal and heuristic sources of knowledge. In this paper we have made an attempt at identifying, representing, organizing and making inferences from these diverse sources of knowledge. We have addressed the issues of separating geometric composition, mechanical behavior, functional models, and explicit reasoning about behavioral assumptions. We stressed the importance of a behavior model from which numeric models are generated and into which numeric results are mapped. We propose the blackboard control architecture as a global organization scheme for domain and control knowledge. Full evaluation of these ideas can not be made until the development of a prototype is completed.

References

[1] James Bennett et. al., *SACON: A Knowledge-based Consultant For Structural Analysis,* Technical Report STAN-CS-78-699, Stanford Heuristic Programming Project, 1978.

[2] Jonathan Cagan and Victor Genberg, "PLASHTRAN: An Expert Consultant on Two-dimensional Finite Element Modeling Techniques," *Engineering with Computers,* 1987.

[3] William J. Clancey, "Heuristic Classification," *Artificial Intelligence,* Vol. 27,1985.

[4] Johan de Kleer, "Problem Solving with the ATMS," *Artificial Intelligence,* Vol. 28, 1986.

[5] Steven J. Fenves, "A Framework for Cooperative Development of a Finite Element Modeling Assistant," *Reliability of Methods for Engineering Analysis,* K. J. Bathe and D. R. J. Owen, Ed., Pineridge Press, Swansea, U.K, 1986.

[6] B. L Gregory and M. S. Shephard, "Design of a Knowledge Based System to Convert Airframe Geometric Models to Structural Models," *Expert Systems in Civil Engineering,* C. N. Kostem and M. L Maher, Ed., ASCE, 1986.

[7] Barbara Hayes-Roth, "A Blackboard Architecture for Control," *Artificial Intelligence,* Vol. 26,1985.

[8] Marie Reynier, "Interactions between Structural Analysis, Know-How and Chain of Reasoning used by the CARTER Expert System for Dimensioning," *Reliability of Methods for Engineering Analysis,* K. J. Bathe and D. R. J. Owen, Ed., Pineridge Press, Swansea, U.K, 1986.

[9] Ian C. Taig, "Expert Aids to Finite Element System Applications," *Applications of Artificial Intelligence to Engineering Problems,* D. §riram and R. Adey, Ed., Springer-Verlag, 1986.

[10] Michael P. Wellman, "Reasonning About Assumptions Underlying Mathematical models," *Coupling Symbolic and Numerical Computing in Expert systems,* J. S. Kowalik, Ed., North-Holland, 1986.