Carnegie Mellon University
Research Showcase

Department of Civil and Environmental Engineering

Carnegie Institute of Technology

1-1-1993

Integrating spatial and functional data in a prototype solids grammar of tall building design

Steven Meyer
Carnegie Mellon University

Steven J.(Steven Joseph) Fenves

Carnegie Mellon University.Engineering Design Research Center.

Follow this and additional works at: http://repository.cmu.edu/cee

Recommended Citation
Meyer, Steven; Fenves, Steven J.(Steven Joseph); and Carnegie Mellon University.Engineering Design Research Center., "Integrating spatial and functional data in a prototype solids grammar of tall building design" (1993). *Department of Civil and Environmental Engineering.* Paper 51.
http://repository.cmu.edu/cee/51

This Technical Report is brought to you for free and open access by the Carnegie Institute of Technology at Research Showcase. It has been accepted for inclusion in Department of Civil and Environmental Engineering by an authorized administrator of Research Showcase. For more information, please contact research-showcase@andrew.cmu.edu.

NOTICE WARNING CONCERNING COPYRIGHT RESTRICTIONS:
The copyright law of the United States (title 17, U.S. Code) governs the making of photocopies or other reproductions of copyrighted material. Any copying of this document without permission of its author may be prohibited by law.

Integrating Spatial and Functional Data in A Prototype Solids Grammar of Tall Building Design

S. Meyer, S. Fenves

EDRC 12-58-93

Integrating Spatial and Functional Data in A Prototype Solids Grammar of Tall Building Design

Steven Meyer & Steven J. Fenves

Abstract

Spatial grammars and solids modeling have both contributed to the expansion of computer reprcsentable abstractions from numerical and symbolic data into the realm of spatial data. Spatial grammars, such as Stiny's shape grammars, have focused on representing and transforming this spatial data. However, to date spatial and non-spatial data have not been fully integrated into an engineering design process. We describe a grammar which uses a solids modeling representation with object-attribute-value labels. The objects in this label data structure are any of the topological elements of the solids model. Through this enriched representation we demonstrate the extensibility of the grammar formalism into engineering disciplines bound by functional constraints. The grammar presented is a prototype implementation intended to demonstrate the relevant domain variables as well as the nature of their interaction in driving the design process. We describe the grammar and discuss the lessons we have learned from its development and critique.

1 Introduction

The introduction of spatial representations has had a large effect on computer tools developed for domains which are highly spatial in nature. In architecture, mechanical and structural engineering the use of spatial information has opened the door to new representations and new computational processes [Stiny 75, Finger 89, Baker 89]. However, while spatial and non-spatial information is tightly interwoven in the building design process this integration has not been reflected in engineering design systems. The spatial relationships embodied in a building configuration and the dimensions of individual members share their importance with material properties and construction methods, yet computer representations of structural systems have relied on incomplete spatial representations. On the other hand, solids modeling research has reached a maturity that begs its integration into computer aided design tools. This report focuses on a prototype solids grammar developed to test initial ideas on integrating spatial and non-spatial information for representing design objects, and for driving a design process, in the domain of tall building design.

The central task of the reported work is the development of a three-dimensional grammar that generates a language of structurally and architecturally valid tall buildings. The grammar concentrates on generating the tube structures of Fazlur Khan as described in Section 2.2. An essential subgoal of generating syntactically valid buildings is the formulation of a vocabulary and syntax of labels composing a well-formed system of operations on functional attributes. As discussed in Section 3, one of the central questions of this research is whether the representation of syntactic structures in engineering design can be enriched to the point where the syntax provides an adequate descriptions of designs without any reliance on semantic interpretations. In this research syntactic structures are composed of three-dimensional shapes and their functional attributes.

The design process of interest is one in which both the architectural and structural considerations of a tall building interact cooperatively. Therefore, the attributes within the grammar must express both architectural and structural functionality at appropriate levels of abstraction. The solids and the functional attributes,

°This work has been supported by the Engineering Design Research Center, an NSF Engineering Research Center.

in conjunction, must be able to support the type of computations which can ensure that the generated solids model will map to a valid, realizable, structural configuration. The representation of functional attributes within a solids model also provides a means for coupling computation and visualization during the design process. The visual and textual presentation of partial solutions along with the user selection of transformations provides both the information and the means for the human designer to participate in guiding the search process. Design knowledge in the form of conditional rules is used to describe the operation of successively transforming the design state from the design requirements to a solution. The construction of a rule base for a domain assumes that the representation for the rules can sufficiently express the spatial and non-spatial attributes necessary for describing the design object itself, the conditions needed to trigger the activation of a rule, and the attributes used in the computation of the new design state. If the domain operations must be triggered by non-spatial attributes of the partial design, the representation must adequately express those non-spatial attributes.

We introduce this grammar by explaining our motivation and purpose before giving a short methodological background. Next, we describe the prototype through a more detailed description of the representation, the design process, and three example designs generated by the grammar. Finally, the discussion and conclusions sections describe lessons we have learned from the development of this grammar.

1.1 Motivation

Spatial design is a primary focus of both the structural engineer and the architect, two of the major participants in building design. Therefore, the construction of two- and three-dimensional spatial models is central to this design process. Current modeling systems provide operations for generating and manipulating primitive geometric elements or for generating objects with a predefined semantic content such as doors or columns. With these modeling systems, a structural system must be generated element by element with little or no higher level assistance. One motivation for investigating the grammar formalism is to seek an environment for valid high-level spatial operations for the generation of structural systems within a design system.

The topologic and geometric validity of a generated spatial model is a necessary but not sufficient condition for the validity of the structural system model in terms of its engineering functions. Additional constraints must be employed during the design process to ensure functional validity. Thus, the use of engineering attributes within transformations is integral to the generation of a configuration appropriate for its function. This approach begs the question of the primacy of form over function, or syntax over semantics. The assurance of syntactic validity is a simpler and more efficiently addressable problem than semantic validity, even though the division between syntactic and semantic attributes is not always clear. Another motivation for the prototype presented here is to formalize a vocabulary and syntax of labels composing a well-formed expression of engineering attributes to be used in conjunction with the solids within the design process. Ideally, the grammar formed of these solids and functional attributes will ensure that the generated solids model can map to a valid, realizable structural configuration.

The development of a prototype is an experiment testing the completeness of our representation and organization of domain knowledge. We recognized at the beginning of this project that it is only through formalizing the details of the domain that we could recognize any "missing links" in our domain knowledge. Also, the adequacy of our representation is only testable by evaluating its ability to abstract the necessary domain knowledge and to describe the design states sufficiently for operations within the design process. As a corollary, we also wanted to know exactly what spatial and non-spatial variables need to be included in the design states to adequately describe a partial solution and to perform the necessary operations.

1.2 Purpose

The grammar reported here is a portion of a larger research project aimed at demonstrating the extensibility of the grammar formalism into engineering disciplines. This extensibility is achieved through the expansion of the simple labels used by previous spatial grammars into a formal system of functional attributes, thereby demonstrating the use of the grammar formalism in domains bound by functional constraints. In finer detail, the purposes of the grammar are to address the issues of high-level operations within a spatial design system for generating designs of tall buildings, the necessity of a formalism for functional attributes within the design system to ensure the functional validity of the generated model, and the composition of a design process model incorporating spatial and functional attributes combining high-level architectural and structural preliminary design. Together, the results of the investigation of these issues will form the components of a spatial and functional grammar of tall buildings. Additionally, a design study was undertaken following the prototype's development to ensure that the knowledge incorporated into the design model and the implementation accurately reflect current practice.

The purpose of the grammar itself is to describe the language of tube structures based on the constructive mode of understanding design [Stiny 78a]. This method of understanding a design style is divided into three objectives: producing existing designs, producing new designs in the style, and clarifying the underlying logical organization of the style.

Producing the existing **designs of Fazlur Khan.** The prototype grammar is written with the aim of generating solids models of the tube structure designs of Fazlur Khan. Due to the complexity of these designs, this domain is a proper testing ground for the investigation of the broader issues involved in this research. One measure of the validity of the grammar is how accurately, and to what level of detail, the grammar does in fact generate models of the extant designs of Fazlur Khan.

Producing **new tube structure designs.** One of the characteristics of grammars is that they provide the compositional machinery to generate new instances of a language. Thus, a test of the grammar is the analysis of its output to determine whether the grammar can generate valid new instances of the design language. The analysis of a generated design can be based on the design's functionality or on the stylistic aspect of the design. The functionality may be analyzed through classical analysis techniques such as matrix methods or a finite element analysis to determine whether the member forces are within the allowable material strength limits, and how closely the axial stress distribution in columns of the perimeter faces approximates the stress distribution of an ideal cantilever tube structure. The stylistic analysis may be performed by a comparison with prototypical features and compositional consistencies of the extant designs in the style.

Clarifying **the underlying compositional structure.** The pedagogic purpose of a design grammar is its clarification of the underlying compositional structure of the design language. The compositional organization is described by the sequential or hierarchical phases of the design process formed as collections of similar rules, or productions, and by the individual transformations within the phases. The grammar of tube structure designs using a design process model incorporating both architectural and engineering function should clarify the detailed development of design solutions. In particular, the grammar should clarify the cooperative interaction of architectural and engineering information in the form of contextual and transformable elements of productions within the grammar.

The primary tasks to be addressed during the development of the grammar include:

- Learning what attributes are relevant to the domain of tall building design, and how these attributes may be divided into spatial and non-spatial classes. Additionally; the spatial and non-spatial constraints on the values of these attributes must be defined in order to define the validity of resulting designs.
- Formulating high-level operations for the generation of a labeled solids model of a building. The organization of high-level operations includes formalizing when and how the relevant attributes are transformed.
- Implementing the attributed spatial grammar, and the evaluation of both the resulting process and the resulting designs.
- Evaluating the syntactic richness of the resulting attributed spatial grammar to determine the necessity for an additional semantic descriptor system.
- Providing a prototype grammar to present to the subjects of a knowledge acquisition study so that experienced designers may critique an existing design system.

2 Background

Structural design is concerned with the description of artifacts whose performance is largely based on their geometric form and the materials used in realizing that form. The specification of a building, whether through drawings or through descriptions of prototypical features such as 'a steel braced frame of three stories with 25 foot bays/ is also in terms of configuration and materials. However, for most interesting structural design problems the number of potential spatial configurations and the complexity of the constraints which must be satisfied make the problem intractable without the careful guidance of search techniques. For this reason structural designers have used, either explicitly or implicitly, consciously or unconsciously, many of the search techniques classified in the artificial intelligence literature including case-based reasoning, goal decomposition, generate-and-test, and constraint satisfaction. To make the problem tractable, the search space and the spatial information is abstracted. Thus, an important area of research is that of defining representations of geometry and spatial relationships, and algorithms for computations on these representations; the area of spatial reasoning. The research projects discussed in the methodological background section are of particular methodological interest because their problem definition is related to the problem addressed by our prototype. The domain background section describes the design problem we are addressing and the characteristics that differentiate it from the design problem of previous projects.

2.1 Methodological Background

A few of the known computer-assisted design tools which serve as a framework for testing design theories are described below. We briefly describe them first and then discuss their implications for addressing our problem domain. Of particular interest is their spatial representation, integration of spatial and non-spatial attributes of the design state, and the representation's impact on the design process.

2.1.1 HI-RISE

HI-RISE [Maher 85] is an expert system for the preliminary structural design of high-rise buildings. The design methodology of HI-RISE consists of goal decomposition and constraint satisfaction. Input to HI-RISE consists of spatial information—overall plan dimensions* number of stories, location and dimensions of the service core—and functional constraints—intended occupancy; lateral loading, and live loading. The output of HI-RISE consists of descriptions of structural schemes in terms of system type, material, bay size, and component size. HI-RISE uses the simplified building geometry of an orthogonal three-dimensional tartan grid [Fenves 76]. Each structural element is associated with a schema instantiating its location within the grid. The representation allows for reliable inference of feasible structural types and calculations of element sizes, as well as the deduction of the spatial orientation and type of an element or subsystem within the grid.

However, the productions in the knowledge-base must be stated explicitly in terms of the predefined object-attribute-value schemas, or must be easily derived from the available attributes in the schema. For example, if the section modulus about the x-axis is not one of the attributes of the steel beam's schema it can not be used in a production because it could not be computed from the spatial model. At a larger scale, the abstracted spatial representation limits the freedom of modifying subsystems of the design by routine geometric transformations such as small translations off of the grid or rotations about arbitrary axes. Thus, the orthogonality assumed by the tartan grid restricts the geometric freedom of design modification. Furthermore, elements are grouped into a strict system-subsystem hierarchy, meaning that each element is a component of exactly one system at a particular level of abstraction. For example, a column in a pair of orthogonal 2D rigid frames must belong to one plane frame or the other, even though in the built structure it is a part of both frames. A more complete spatial model of a design would allow system-subsystem relations to be evaluated from an ever-changing representation of the design state. These examples of the limitation of rigid data structure organizations and restricted spatial representations prompt our investigation of a more complete spatial model which would allow less restricted queries and transformations.

2.1.2 ABLOOS

ABLOOS assists a human designer with layout tasks, such as architectural or circuit board layout by enumerating alternatives and handling possibly conflicting constraints and criteria. More specifically, the layout problem is: given a bounding rectangle, *n* component rectangles, and a set of constraints; find the possible non-overlapping placements of component rectangles within the bounding rectangle. The central process of ABLOOS, then, is the stepwise composition of a configuration of loosely packed rectangles.

The representation used in ABLOOS is a graph of objects organized by the orthogonal spatial relationships between rectangles based on Flemming's orthogonal structures [Hemming 86, Hemming 89a], allowing the generation process to be divided into topological and parametric operations [Hemming 89b]. Each node in the graph represents a *goal objects* or GOBs which may represent a single element or a collection of objects at a common level of abstraction [Coyne 89]. The generation method is based on hierarchical decomposition and a generate-and-test strategy. Hierarchical decomposition uses GOBs as representative of a complete layout at a particular level of abstraction, and the generate-and-test strategy places an object and evaluates the resulting layout with respect to the applicable constraints. The inputs are an ordered list of goal objects to be laid out within a prescribed geometric space and a collection of constraints and criteria associated with specific component types. Two classes of constraints are successively employed: topological constraints which limit the spatial relationship between GOBs and spatial constraints which restrict dimensional properties (e.g., maximum or minimum area of a rectangle). The topological description

specifies a class of solutions containing all the geometric instantiations. Output is a graph describing the topology of the configuration and the geometry of individual GOBs.

This approach is exemplary in abstracting the topological and geometric aspects of a rectangular dissection in order to develop a formal representation of the topological aspect as a graph whose nodes represent the rectangles and whose directed arcs represent one of the spatial relationships. A well-formed solution is assured by proving that the representation is closed and complete under the application of a small set of generative rules. Thus, the representation is used to prove theorems about the solutions, placing the approach on a firm theoretical basis. However, two aspects of orthogonal structures limit their use in representing the structural systems of buildings. First, the rules for generating orthogonal structures are embedded in a design method which requires knowing the precise number of objects to be used, and these objects are added one at a time while meeting a set *of a priori* adjacency constraints. In a structural system the number of elements employed may change from one potential solution to the next; the exact number of elements is rather unimportant. Likewise, the adjacency requirements on the elements of the design in our domain are a result of the partial solution rather than a part of the problem statement. Secondly, we would like to be able to employ elements whose edges are not necessarily parallel to an orthogonal grid. That is, we would like to be able to use diagonal elements such as those in trusses or braced frames. Therefore, we are inspired by orthogonal structures and their formation of the basis of a design method, but we are searching for a representation more appropriate to our domain.

2.1.3 Shape Grammars

A shape grammar is a formalism for defining algorithms that operate directly on shapes, labeled shapes and parameterized shapes, where a shape is a finite collection of maximal lines [Stiny 80]. The algorithm embodied in the grammar procedurally defines a language of related shapes—the design space of the domain [Stiny 78b, Koning 81, Renaming 81]. The shape grammar formalism employs a two-part representation. Spatial information is primarily carried by the shape portion of the design state while non-spatial information is carried by labels; symbols located at geometric points associated with the shapes[1]. Thus, we talk about the vocabulary of a shape grammar being composed of shapes and labels.

In a labeled shape grammar the shapes are used to represent the design object itself. A finished design contains no labels because the labels comprise the non-terminal symbols of the vocabulary. Labels are temporary attributes of a design state that may be used to distinguish phases in the design process or to locate a rule's application within a set of shapes. Both uses restricts the rule's application to a subset of shapes within the vocabulary. This is necessary because a simple shape is typically not a rich enough representation of a design to property constrain rule matching through spatial attributes alone on an otherwise syntactically ambiguous portion of the geometry. The use of labeled points to denote a phase in the design process imposes an ordering of potential rule applications and simplifies the matching process when transforming geometrically simple subparts of a design. Labels are also used to define termination conditions; in the usual shape grammar formalism generation terminates when no labels are present. Thus, a label is a temporary mark which is absent in all terminal configuration. Using labels in this way, as a separate algebra, divides them from shapes in that shapes are a representation of the abstract model whereas labels are organizational conveniences with no counterpart in the physical world. Thus, labels act as global or local constraints to aid syntax, efficiency or user-readability.

---

[1]A shape grammar label is defined as an ordered pair $< p,A >$ where p is a geometric point in the same coordinate system as the shapes and A is a symbol label, typically a short string of one or more letters.

More complex constraints have been incorporated into shape grammars to limit the application of rules based on logical properties of the domain rather than purely on the syntactic content of the design state [Krishnamurti 78]. These constraints are specified as predicates associated with particular rules. The predicates are tested against labels in the design state, rather than simply matching on the existence of labels, beginning the formalization of computations on labels within a rule's matching and transformation. However, this is an atypical use of labels; the predicates are used to restrict the grammar of a general domain—plan layout—to more specific domains depending on the constraint set (e.g., Palladian plans, 2-rectangulations, etc.).

Research on the use of the grammar paradigm of design has largely focused on algorithms for shape operations and has largely ignored the formalization of operations on the label portion of the representation. This may be understood as either the perception of a greater importance for the spatial aspects of design, or the greater generality of spatial algorithms. Algorithms operating on the spatial aspects of a design are domain-independent whereas non-spatial aspects of a design and design process are highly domain-dependent and semantically driven. The algebra of shapes that governs their transformations is a purely syntactic system operating without regard for the meaning of the objects being represented. In contrast, a computational system for labels must have a domain-dependent basis for ensuring the validity of operations transforming labels and coupling the values of labels with the parameterization of shapes. A domain such as structural engineering, which is constrained by non-spatial aspects of a design state, relies on functional information such as material property data to help drive the design process. The importance of non-spatial aspects of the design state questions the sufficiency of extending the grammar formalism simply by partitioning the labels into a nonterminal and terminal vocabulary to allow labels in a finished design state. As mentioned in Section 1.2, the procedural definition of a language of design is intended to clarify the underlying structural organization of the language through its organization of the compositional machinery able to produce that language. The description of the organization of design processes in engineering domains must be expressive of the non-spatial aspects of this organization, not merely its spatial aspects. Thus, the nature of engineering domains raises two questions about extending the grammar model of the design process to generate languages of designs that are constrained by both spatial *and* non-spatial aspects of the domain.

1. Does the description of a functionally constrained domain through the grammar model require the specification of the computational basis for the transformations of the non-spatial aspects of the design and their coupling with the shape algebra?

2. Is a syntactic specification sufficient or must a semantic system be composed to guide the transformations and their coupling with the shape operations?

The current research is founded on extending the grammar model of the design process to functionally constrained domains by addressing these questions.

2.1.4 Solids Grammars and GENESIS

The grammar paradigm evolved into three-dimensions with the development of GENESIS, a boundary representation solids grammar interpreter [Heisserman 90]. Like shape grammars, a boundary solids grammar is a rule-based formalism for generating primarily spatial models of a design. Unlike in shape grammars, these models are rigid solid objects represented by their boundary elements: vertices, edges and faces, and by geometric information associated with each of the vertices. Non-spatial information may be associated with any of these boundary, or topological, elements using labels having an object-attribute-value structure. Rules

are used to match on aspects of the solids and labels in the current state of the design, and then to modify the solids and labels or to create new ones. Like a shape grammar, a solids grammar consists of a terminal and nonterminal vocabulary; an initial, or axiom, state consisting of a set of labeled solids; and a set of rules. From this specification, a solids grammar can be used to produce a language or family of solids as designs. The assurance of a geometrically valid solid is accomplished through reliance on the Euler operations for topological transformations. Again, the spatial algebra is formalized and extended, while basically ignoring the question of the fonnalization of a computational mechanism of non-spatial attribute transformations. Like shape grammars, the solid grammar formalism allows the developers to devise any design process they wish. However, there is no impetus from the grammar paradigm to employ any particular design approach such as goal decomposition or prototype refinement, nor is there any built in support for managing such strategies.

The availability of working grammar interpreters is limited. Essentially, the only working three-dimensional solids grammar interpreter is GENESIS [Heissennan 90] which uses a split edge data structure for its solids modeling representation. This representation covers the domain of geometric solid objects, but we were concerned about the low level at which one must program a rule set.

1. Could a reasonably small set of primitive functions be written that would allow the partial design solutions to be manipulated at appropriate higher levels of abstraction?

2. Are collections of atomic object-attribute-value data structures, even when part of a formal system, sufficient to express all relevant non-spatial aspects of the design and the design process?

3. Are the objects of the object-attribute-value structures always complete solids that represent complete physical objects, or are labels required to denote elemental portions of solids such as faces or vertices?

These questions are a primary motivation for the development of the prototype reported here.

2.1.5 Primitive **Hut** Grammar

Mitchell, et al. adopt the generate-and-test strategy for design by integrating a spatial grammar as the generator with analysis procedures as the tester [Mitchell 91, Mitchell 90]. They demonstrate their framework on a primitive hut grammar that generates a single room rectangular building with a flat or peaked roof. The form of the primitive hut is also given a structure with sufficient strength to withstand gravity and lateral static loads. Their grammar adopts a hierarchical strategy, first transforming the initial object into 'marker' objects which abstractly represent the objects' location and functionality and then transforming the markers into more refined markers and finally into specific objects or subsystems that fulfill that functionality it a particular manner. The markers are the nonterminal symbols of the vocabulary and the terminal vocabulary is composed of subsystems, such as trusses or sets of trusses, and elements, such as columns or braces. All elements of the vocabulary are parametric objects. The parameters, or dimensions and divisions, are specified by a tree of dimensional dependencies that follows the top-down parts hierarchy. When no more marker objects are present in the design state, the design is a well-formed input to the testing phase.

The analysis procedure applies elementary structural mechanics formulae to the terminal objects. The results of these formulae are shown to the user as colorings of the members in stoplight fashion: red members are improperly sized, green members arc property dimensioned and yellow elements arc at the edge of structural adequacy. The user may then adjust the dimensions of the members using a sliding scale in the graphic user interface. Thus the terminal elements arc actually the non-markers with green colorings.

However, it is theoretically possible that the alteration of green members can turn red members to green, but the alteration of terminal elements is a contradiction. This raises a few questions about the integration of grammars and analysis, and the adherence to the grammar model.

- Are green members changeable? If so, are they terminal elements?
- If the grammar is not being used during redesign can we call this an integration of a generative grammar and analysis? And if the grammar is being used during redesign is the alteration of parameters specified by productions on the shapes?
- The design variables described by Mitchell, et al. are all spatial attributes, yet it is clear that some material considerations are included in the object description and during analysis. How are these non-spatial aspects represented and integrated into the grammar?

These questions will be answered by gathering further information on this project.

2.2 Domain background: The tube structures of Fazlur Khan

From the inception of the tall building at the beginning of this century the traditional construction method has been the beam-column frame system. Semi-rigid frames and braced frames have had a limited place in the design of the smaller high-rise buildings, whereas rigid frames have applicability over a wide range of heights. The Empire State Building, for example, was built to a height of 120 stories in the early 1930s using a steel rigid frame. In the rigid frame the beams are rigidly connected to the columns so that the lateral loads are resisted by bending in the beams and columns. Yet, for buildings with large height to width aspects, rigid frames may produce excessive deflections due to the bending of these members. More importantly, present tall building design and construction conditions differ from earlier conditions in three respects. First, larger column spacings are expected in modem office buildings. When the Empire State Building was built a column spacing of 20 feet was acceptable. Today, office column spacing is expected to be 40 feet or more. Second, interior partitions are constructed as temporary elements of a building today, and hence cannot be expected to contribute to the rigidity of the structure. Third, the exterior cladding of modem glass curtain walls, as opposed to the stone cladding of earlier buildings, also does not add to the rigidity of the structural system. Due to these characteristics of earlier tall buildings, their actual lateral drift was often markedly less than the computed value used in design. However, in contemporary construction the frame must stand entirely on its own and calculated drifts are close to the building's actual performance.

The sizing of members in a frame structure is guided by two factors: gravity loads and their effects and lateral loads and their effects. Given a building's dimensions, the gravity loading effects cannot be eliminated through altering the placement or sizing of members. However, the results of lateral load effects *are* affected by the structural configuration. In a rigid frame structure, the lateral drift is itself a sum of three factors: bending moments in the girders contribute almost two thirds of the total deflection; bending moments in the columns contribute almost one sixth, and axial forces in the columns due to overturning moments contribute about one fifth [Khan 67]. The first two effects represent the frame action while the last effect is based on statics and thus cannot be avoided. If the plane frames were replaced by infinitely stiff plates the frame action could theoretically be eliminated. If this rigidity could be achieved in practice, then lateral drift would be reduced to about one fifth of a comparable frame. This rigidity can be approached by structural systems which, rather than acting as rigid frames, act like rigid boxes or tubes composed of the

perimeter faces. The structural tube eliminates the need for increasing the column and girder sizes in order to reduce lateral drift due to frame action, which in turn reduces the premium for height.

One way of producing a frame which simulates a rigid tube is by replacing all the exterior columns with diagonal columns in both directions in the plane of the building's face. If these columns are spaced closely enough they will simulate a bearing wall. When the columns in adjoining faces are connected at the corners the system acts as a rigid tube. This 'diagonalized tube' most closely approximates the character of a cantilever. But since the gravity loads are also to be resisted by the inclined columns, the size of these columns must be increased in proportion to their angle, resulting in overly large columns. Another method of approximating a rigid tube is through the use of very closely spaced exterior columns which are connected at each floor by deep spandrel beams. The 'framed tube[9] is analogous to a shearwall system placed around the perimeter of the building rather than around the service core. A combination of these two systems is called the 'trussed tube[9] using both vertical and diagonal columns. By connecting the vertical columns and the diagonals the vertical loads are redistributed throughout the face resulting in an equivalent rigid bearing wall. This can be achieved even though the exterior columns are spaced widely apart.

In practice the tube system acts partially as a true cantilever and partially as a beam-column frame system. The overturning moment under lateral loading is resisted by the tube form causing tension and compression in faces perpendicular to the loading, whereas the shear is resisted by bending in the frame of the faces parallel to the lateral load. The perimeter faces of the structure are idealized as the walls of a hollow tube. But because the faces parallel to the lateral force—the webs of the hollow tube—are not actually solid webs but are instead frames, these webs lose some of their rigidity due to bending in the framed web. This " shear lag* results in a distribution of forces as shown in Figure 1. As can be seen in the figure, the shear lag also occurs in the faces normal to the lateral force—the flanges of the hollow tube—because the in-plane flexibility of the flange prevents the development of a uniform 'stress[9] distribution in the flange.

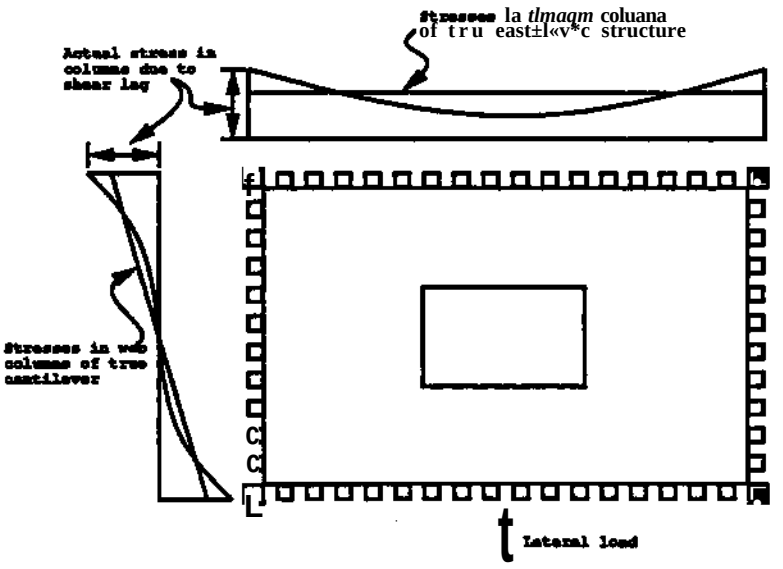


Figure 1: Shear lag in a building subject to lateral loading.

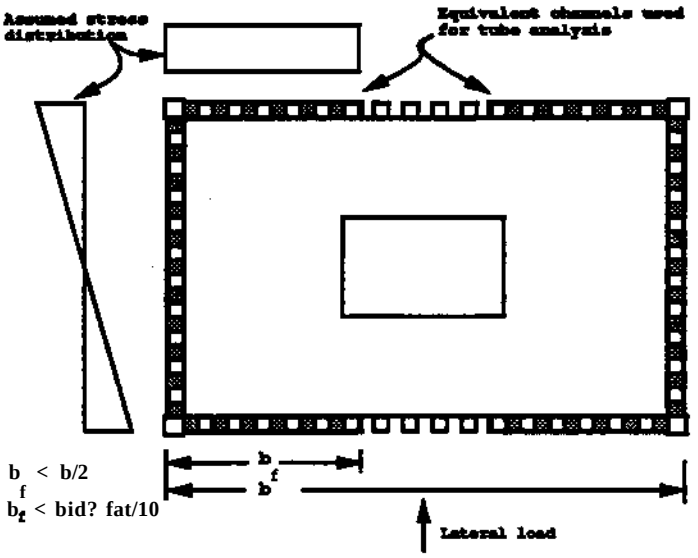


Figure 2: Equivalent channel pair for the preliminary analysis of tube structures.

For preliminary analysis, the tube is considered as two equivalent channels whose webs are parallel to the lateral loading and whose flanges total less than the width of the structure, (see Figure 2) [Khan 73]. An analysis of these channels using then gives maximum values for shear and moment of the modeled structure. Then, these computed forces can be used to dimension members of the proposed structure. Because the tube structural system relies on the interaction between frames orthogonal to the direction of the lateral loading *and* on frames parallel to the direction of loading, the design process must consider the three-dimensional nature of the mechanism.

From this cursory description of a preliminary analysis procedure, it is clear that the design process must consider the structure as a three-dimensional system. A frame structure, in contrast, resists lateral-load induced moments and shears through bending in plane frames parallel to the direction of the loading and can be designed as a two-dimensional system. Despite the complex analysis required in its design, the tube structural system makes possible the economic construction of tall buildings by using the rigid connections in the perimeter walls as the building's lateral bracing, alleviating the need for internal bracing which is both expensive and reduces flexibility in architectural planning. The tube structure in its various forms has been used in both steel and concrete framing systems to reduce overall structural weight and, therefore, structural costs.

This discussion of previous representations of design states and models of the design process along with the discussion of domain requirements demonstrates the complexity of the tasks we are addressing. The next sections describe how we are addressing these problems. First the representation used in the prototype is described. Section 5 describes the design process using this representation, and Section 6 presents a design example generated by the grammar.

3 Representation

The current practice of using two-dimensional drawings to describe a building succeeds because humans (frequently) can combine the two representations into a three-dimensional mental model. The prototype grammar is an investigation of the expressiveness and expediency of a design system based on a representation integrating a three-dimensional solids model and object-attribute-value structures. The examples of grammars described in Section 2.1 differ in their spatial representations and in their representation of non-spatial attributes. In the prototype grammar we have followed the shape and solids grammar formalism except for two aspects of the label vocabulary. First, we allow, or rather we rely on, the use of persistent labels to describe the non-spatial attributes of a finished design. That is, the terminal vocabulary contains labels used to describe non-spatial aspects of a finished design. Secondly, we use an object-attribute-value triplet for the representation of labels rather than the object-value pair of labeled points as in shape grammars.

The representation of a design state in the prototype grammar consists of a solids model and object-attribute-value labels attached to any topological entity of the solids including the solid itself[2]. Solids are represented as boundary representation solids models using the split-edge data structure. This boundary representation of solids provides an complete, unambiguous representation. All non-spatial attributes are represented as labels attached to the solids using an object-attribute-value syntax, and more than one label may be attached to a single topological element of a solid, thereby forming object-attribute-value structures of functional attributes. Thus, all design states are represented by a uniform coupling of two abstract data types: a set of three-dimensional planar-faced solids and a set of object-attribute-value triplets. The state representation is composed as the product of the vocabularies formed from these two data types.

The expressive advantage of the object-attribute-value triplet over an object-value pair is in the addition of a metric context to the value. The joke about giving baseball scores without telling the teams illustrates the advantage of the object-attribute-value structure, allowing the labels to describe what the "value" is a value for. When we are concerned about basically spatial aspects of a design the object-value representation may be sufficient for orienting the application of a rewriting rule, but when we are concerned with completely non-spatial aspects of a design an object-value representation is insufficient for expressing the context of the value. For example, in Hemming's Shadyside grammar [Hemming 88] object-value labels are used to describe the orientation and location of a rectangle representing a room. A pair of Fs and a pair of Bs at two pairs of adjacent comers of the rectangle signify the front and the back of the house, respectively. The four labels are used to locate the application of rules on an otherwise syntactically ambiguous portion of the geometry. Without these labels the rectangle is not a rich enough representation of a room to property confine the grammar through spatial attributes alone. However, to represent functional attributes such as allowable loading or material properties, the use of a geometric point associated with a quantitative value would lead to ambiguities as to what those values stand for. For example, if allowable loading in pounds per square foot is 50 and the allowable yield stress in kips per square inch equals 36, then an object-value label representation would lead to two points labeled SO and 36. The only technique for distinguishing the semantics of the two labeled points would be their location, because the loading could be 36 pounds per square foot, and the allowable yield stress could be SO kips per square inch. The values themselves are not arbitrary and therefore when their ranges coincide they are ambiguous. By adding another field to the label representation both the designer and the machine can use the attribute field to distinguish between two labels representing different non-spatial attributes.

[2]The topological elements of the solids model form a hierarchy composed of the following elements listed in top-down order solids, shells, faces, loops, edge-halves, and vertices.

Thus, a design state 7 is represented by a set of solids and a set of object-attribute-value structures:

$7 = \{S,A\}$ where S : A set of elements of valid three-dimensional solids.
A : A set of object-attribute-value structures, where the objects are topological elements of S.

The rewriting rules that operate on this state representation are implemented as collections of PROLOG clauses on both the left hand, or condition, side and the right hand, or action, side of the productions. Each clause refers to topological entities of the solids, to the labels attached to those entities or to both. The production's condition is composed of an inclusive and an exclusive portion, each of which contains a set of elements of solids and a set of object-attribute-value structures. As many as three of these four portions of the condition—inclusive solids, inclusive attributes, exclusive solids, exclusive attributes—may be the empty set. The production's action may contain elements in any of these four sets which are not transformed by the production, comprising the context in a context-sensitive grammar. The action side of each production contains a set of new solids or modifications to the existing solids, and a set of new object-attribute-value structures or a modification of the existing labels. Either or both of these sets may the empty set. More specifically, a production p is represented as:

$$\gamma \underset{p}{\Rightarrow} \gamma'$$

$$\{S_{incl}, A_{incl}, S^*_{incl}, A^*_{incl}, S^*_{excl}, A^*_{excl}, S_{excl}, A_{excl}\} \wedge \{S_{incl}, A_{incl}, S_{new}, A_{new}, S_{excl}, A_{excl}\}$$

where

S_{incl} : A set of solids, or elements of solids present in the current model which are unaffected by the production.
S_{incl} *€7 and SuuiEy'.*

A_{incl} : A set of object-attribute-value structures present in the current model which are unaffected by the production.
A_{incl} *e* 7 and $A^\wedge$ *e 7'.*

S_{excl} : A set of solids, or elements of solids present in the current model which are unaffected by the production,
$S_{excl} \cap \gamma = \emptyset$ and S_{exc}j n 7' = 0.

A_{excl} : A set of object-attribute-value structures which must not unify with those present in the current model which are unaffected by the production.
*An** n 7 *m* 0 and A_{wi} n 7' = 0.

S^*_{incl} : A set of solids, or elements of solids present in the current model which are removed from the model by the production.
$S^*_{incl} \in \gamma$ and $S^*_{incl} \cap \gamma' = \emptyset$.

A^*_{incl} : A set of object-attribute-value structures present in the current model which are removed from the model by the production.
$A^*_{incl} \in \gamma$ and $A^*_{incl} \cap \gamma' = \emptyset$.

S^*_{excl}: A set of solids, or elements of solids which must not be present in the current model.
$S^*_{excl} \cap \gamma = \emptyset$ and 5 ^ e S_m *6 7'.*

A^*_{excl}: A set of object-attribute-value structures which must not unify with those present in the current model.
A ; ^ n 7 = 0 and A ^ e A™ € y.

*Sn*w*: A set of solids, or elements of solids, generated by the production.

Aw : A set of object-attribute-value structures generated by the production.
$A_{incl} \cup A_{new} = A' \mid \gamma' = \{S', A'\}$.

Thus, each side of the production is composed of two compound predicates, one predicate on contextual labeled solids and one predicate on labeled solids to be rewritten. The contextual predicate contains inclusive and exclusive clauses which are true before the production matches on the state and remain true after the rewriting takes place. The rewriting predicates contain inclusive and exclusive clauses on the left-hand side of the production, but not true afterwards. In this way a context-sensitive grammar is represented using predicates on labeled solids.

The inclusive predicates in the condition of a production typically test for the presence of labeled solids to be rewritten by the production. Alternately, the inclusive and exclusive predicates may test for the presence of sufficient contextual information to perform an operation transforming the labeled solids in the current state. The shapes and labels unified in the production's condition may be required for calculating variables in the parametrically defined solids and labels. Alternately, the presence of sufficient information can signify that an appropriate phase of the design process has arrived for performing the production's transformations, or the contextual information may represent additional constraints on spatial and non-spatial aspects of the design state that is being transformed.

A frequently employed use for labels on spatial elements is the labels' enhancement of the computational efficiency of matching on specific labeled solids in the design state. For example, topologically or geometrically similar objects may be indexed by a label to aid in the recognition of a particular one of the similar objects. The similar solids of corner columns in the tube structure arc given labels with the same attribute field leaving the value field to hold the variable index value. It would be possible to avoid such labels by collecting and then ordering the set of similar objects, but it would be more computationally expensive and therefore difficult to justify only on the grounds of formal purity.

In summary, the above representation provides for the implementation of productions of a context-sensitive grammar. The representation couples three-dimensional solids with object-attribute-value structures expressing constraints necessary for ensuring the functional validity of the generated building model. The next section describes the computational environment in which this representation was implemented. Section 5 then describes the design processes which make use of this representation before Section 6 presents an example design.

4 Design Environment

This section briefly describes the machine environment in which the grammar and the human designer generate designs. The prototype grammar is implemented using the GENESIS boundary solids grammar interpreter. GENESIS defines a general representation for both the state and the rules for modifying the state, provides the mechanisms for unifying rule conditions with the current state, for instantiating and modifying the solids model and labels, and for displaying the solids in the current model. To accomplish these necessary features of a grammar interpreter GENESIS provides a boundary solids modeler, a database for maintaining labels, a logic programming interpreter/compiler, and a graphic user interface. The solids modeler uses a generalized split-edge representation that allows the modeling of non-manifold solids through the employment of a vertex-use and shell-use structures. The solids modeler also provides Euler operators and geometric transformations for generating and maintaining valid manifold and nonmanifold solids. The label database provides a means of associatingnon-spatial information with any of the topological elements of the solids keyed on any element of the object-attribute-value label structure. Efficient label access is provided by directly indexing the topological elements into the label database. The logic programming interpreter is used for integrating the solids model with the productions of the grammar and any auxiliary predicates,

and as the textual interface to the design state. GENESIS uses IBM's compiler based implementation of the CLP(R) language, a constraint logic programming language over the real numbers [Jaffar 90], for production unification and application. Within the solids modeler primitive topological and geometric unification and transformations are accomplished through built-in predicates. Access to the label database and the graphics routines is also accomplished through built-in predicates that operate directly on the database or solid model, respectively. The CLP(R) compiler itself provides the unification mechanism for matching productions with the design state, and a backtracking mechanism for when partial matches cannot be carried further. The use of a constraint logic programming language has made the matching process a result of the programming language's own unification and constraint satisfaction process. This eased the development of parameterized rewrite rules, since the application of production is based on unification and constraint satisfaction rather than on direct matching. The developer of a grammar can construct complex operations and matching conditions on top of these two mechanisms for use in either the precedent or consequent of a rule. The graphic user interface displays the solids model portion of the current design state, and can highlight portions of the model unified or deterministically selected by the rule. The graphic interface includes a control panel for adjusting the camera viewpoint, light source positions and other graphic features.

The CLP(R) interpreter's text display shows a developer-written description of the first currently applicable production, informing the designer of the result or motivation of a production before asking the user's approval for the rule's application. In this way the designer can make an informed choice in traversing a path through the design space. The productions may also write to the text window to inform the designer about parameterized matches performed in the production's condition and modifications of the design state performed as part of the production's action. Additionally, the text window may be used at any time for querying the design state. These queries can be formulated in terms of basic solids modeling aspects such as "Show me a face whose area is grater than 20 and whose normal faces directly upward." or in terms of high-level predicates specifically written for this grammar, such as "How many floors are there in the apartment volume?" In the later case if there are two apartment volumes, then the query will unify with two different apartment volumes and return two answers.

GENESIS is a self-contained grammar interpreter which may also be linked to other software packages on the same machine, or across a network, to integrate a solids grammar with other processes. This is accomplished by clauses in a production that call compiled programs or shell scripts, features that have been used to advantage in automating the performance of analyses of the design state. A predicate that calls a program or script waits for the called program to finish before the clause succeeds, maintaining a predictable sequential succession of clause trial.

5 Design Process

The use of a spatial grammar to model a design process defines the representation and intention of the model, but offers little assistance by itself for organizing the design process. It may seem natural to employ hierarchical decomposition when developing the grammar, but this is only a guiding principle to be constructed by the grammar's developer. Since all design knowledge is embodied in the shapes and labels of the productions themselves, any organizational or methodological instruments must be introduced as temporary shapes or labels in the design state. In this section we describe our organization of the design process and our emulation of useful design methodologies within a grammar model of tall building design.

The design process employed in the prototype solids grammar is based on hierarchical decomposition, achieved through the successive generation of abstraa labeled solids intended to fulfill an architectural or structural function, and the recursive modification of these abstract labeled solids in order to fit them to the functionality required for the design instance at hand. In a global view, the design approach is the recursive application of four steps which define and transform the solids and labels of the building model. Taking a more localized view, the design process involves the three tasks of setting the initial conditions, generating the architectural volumes and generating a structural system. The latter two tasks are subdivided further into multiple subtasks. This section begins with a domain-independent view of the employed design process, continues with a domain-dependent view of the design process and then describes each step in the domain-dependent process more specifically.

The definitions and transformations involved in each task in the design process may be viewed as the interaction of the following four steps:

1. Object introduction or definition, a subdivision of space.
2. Label introduction or definition, a partition of the set of descriptors of building function into pertinent and non-pertinent sets.
3. Spatial transformation of objects, 'shaping' objects to satisfy spatial and functional constraints.
4. Label value assignment to define or satisfy functional constraints.

Each design task consists of the application of these four steps, generating a representation of the building design at one level of abstraction. However, within an individual design task the application order of the steps may vary depending on the requirements of that particular design task. The two pairs of steps—introducing and transforming solids and labels—are used to separate the decision of using or not using a design element (e.g., a tube or a diagonal brace) from the refinement of that element (e.g., refining the tube into more specific members or dimensioning the diagonal brace). The separation of introduction and transformation steps is used to a lesser degree with labels; no label is introduced into the design state without defining both its attribute and value fields, but once introduced, a label's value field may be repeatedly modified (e.g., member force labels attached to beams, columns, etc; the analysis-done versus analysis-needed flag.)

The object introduction step topologically divides three-space into a portion representing the design state and a portion representing the external environment. Likewise, the label introduction step inserts into the design state an attribute that is deemed important for describing the design state, dividing the set of attributes in the vocabulary into a subset that is descriptive of the current design state and a set that is not. For example, a step introducing the material property labels says that the construction material is a necessary non-spatial attribute of the design at this stage of the design process.

The statement of an attribute's importance in describing the current level of design refinement is a related, but separate, statement from deciding what the value of the attribute should be. Frequently, a step such as material assignment will be satisfied by selecting one production from a set of productions. Grouping these productions together according to the attributes that they introduce organizes the design process according to a structure unaffected by the particular values that the attributes may be given. For example, when the first structural analysis is performed, a label with the attribute field a n a l y s i s - s t a t e is attached to the abstract tube. Before an analysis is requested by the designer, the analysis state is irrelevant. After an analysis is requested, the design state must keep account of the analysis state. The value of this label

describes the correspondence between the recent analysis and the current design state, but the very existence of this label describes the design process as including an analysis that must be updated as the dimensions and material properties are altered.

Tasks and subtasks within the overall design process are delineated by a label attached to the site lamina. The value field of this label is matched on by many of the productions to test if the production applies to the current task or subtask in the design process, regardless of other aspects of the design state. The design process consists of the following ordered tasks:

1. Establishing the initial conditions.
2. Generating architectural volumes.
   (a) Shaping the architectural volume.
      i. Intra-volume transformations,
      ii. Inter-volume transformations.
   (b) Instantiating a service core.
3. Generating structural system.
   (a) Instantiating the abstract tube.
   (b) Refining the abstract tube.
      i. Material selection.
      ii. Tube class selection via abstract member instantiation,
      iii. Member dimensioning.
   (c) Instantiating secondary structural members.
   (d) Analysis and feedback.
   (e) Modifying tube and secondary members in response to analysis.
   (f) Detailing the tube system, e.g. ground floor modifications.
   (g) Looping back to tube member dimensioning when analysis evaluation warrants.

The following sections describe these tasks in more detail. The tasks in this hierarchy frequently reflect tasks within current design practice, and we first describe the relevance of the task in actual practice before describing our emulation of the practical task within the grammar.

5.1 Establishing the initial conditions.

In practice the first stage of a collaborative design process occurs when the owner, architect and structural engineer meet to discuss the initial building program. At this stage the owner has selected a site and performed market research to focus the occupancy requirements of the building. Thus, site and occupancy information form the initial conditions of the practical design process, allowing the architect and engineer to prune the building types under consideration from the complete range of existing building types to only those building types of a scale that meet the owner's needs.

The design process employed in the prototype grammar begins from the site and occupancy information that would be available from the developer and from current design standards. Therefore, the initial solids

of the grammar are a horizontal lamina and a unit cube, abstractly representing the site and the building, respectively. The following attributes and their instance-specific values are labels attached to these initial solids.

- Total site dimensions.
- The classification of the building's geographic location in order to calculate expected lateral loads as a function of building height.
- An occupancy class list partitioning the building into *n* vertical occupiable units[3] associating each occupiable unit with an occupancy class.
- The gross square footage for each occupiable unit.

The last two attributes, describing occupancy and floor area, are attached to the unit cube whereas the first two attributes, describing site dimensions and location, are attached to the lamina. The occupancy and floor area attributes are attached to the unit cube as two lists in the value field of the occupancy-classes and floor-areas labels, respectively. These two lists have the same length which is used to implicitly define the number of occupiable units needed by the designer's client. The site dimension and location attributes are also attached as labels whose values are structured as lists. Site dimensions are given as a list of breadth and depth since the prototype is restricted to rectangular sites. Geographic location for the purpose of calculating lateral loads is defined by a list of the three independent variables of the ANSI building code: basic wind speed in miles per hour, ANSI exposure class and distance from the atlantic or gulf coast for the purpose of calculating hurricane exposure [ANSI 72]. These site attributes are attached to the lamina as two labels whose attribute fields are site-dimensions, and ansi-location. Thus, the lamina and the single unit cube both are given two labels structured as lists and the resulting two labeled solids define the initial, or axiom, state of the grammar.

5,2 Shaping the architectural volume.

As the architect begins deciding on a rough building form, a refinement process is begun. First the architect investigates what rough bulk will fit on the site and meet the owner's space requirements. This rough bulk is less specific than an architectural form, merely having height, breadth and width; the shape of this bulk is irrelevant. After these three dimensions are roughly proportioned the architectural form can be realized.

The approach used in generating the architectural volume is hierarchical. Beginning from a representation of the building model as a lamina, a single cube and the initial set of labels, the unit cube is transformed into a vertical sequence of unit cubes representing the building's separate occupiable units. The sequence of unit cubes is then successively transformed geometrically to suit the dimensional constraints of the building's site and use.

---

[3]An occupiable unit is defined as a continuous, occupiable volume of a building associated with a particular set of functional requirements (i.e$_M$ an occupancy class, adjacency to particular other functional volumes, etc.), and bounded above and below by the building's foundation, the building's roof, or by another occupiable unit An occupancy class is the intended use of an occupiable unit, i.e., one of the set {lobby, commercial parking, general office, executive office, apartments, studio, observatory, restaurant, mechanical floor}. An occupiable unit is associated with exactly one occupancy class, but a particular occupancy class may be associated with more than one occupiable unit All the building designs of Fazlur Khan have contained more than one occupiable unit Even his buildings devoted to apartments have included a lobby volume in addition to the apartments' volume.

The length of the sequence of cubes—the number of cubes stacked on top of each other—equals the length of the occupancy-classes label's value. The production generating the sequence of cubes also introduces four labels attached to each of the unit cubes including the initial cube. These four labels are atomic-valued labels whose attribute fields arc occupancy, square-footage, ideal-lease-span, and floor-to-floor-height. The occupancy and square footage values arc taken directly from the initial cube's listed values, but the ideal lease spans and preliminary floor-to-floor heights arc looked up from a set of CLP(R) clauses indexed on occupancy class. This distribution of the labels that were previously attached to the single unit cube as listed values makes the original list-valued labels obsolete and they arc deleted.

The geometric transformations performed on the occupiable units arc divided into a subtask that only considers an individual occupiable unit's geometry and labels and the site-dimensions label when transforming that occupiable unit, and a subtask involving transformations that consider the inter-relation of the sequence of solids, which arc probably no longer cubes. These intra- and inter-volume transformations arc described in the next two sections.

5.2.1 Intra-volume transformations.

Next, each occupiable unit undergoes a series of transformations *in isolationfrom the other occupiable units.* The transformations in this intra-unit subtask shape the occupiable units to meet constraints specific to that individual unit. Each occupiable unit must first be enlarged from a unit volume and then must be shaped to suit its intended use. The first transformation changes the unit volume to the volume required as the product of the required square footage and the preliminary floor-to-floor height. This transformation is performed as a simple scaling of the cube.

Next, the cube of the proper volume is transformed along the site's shorter dimension so that the occupiable unit has a desired core-to-perimeter dimension, is no larger than the site and still has the proper volume. The horizontal dimension is based on that unit's occupancy-class and ideal-lease-span labels. In the current market, the desired dimension for an office space is a clear span of 40 to 42 feet from the service core to the perimeter. In contrast, current apartments are planned on a 25 foot dimension from inside wall to exterior wall. Adding the interior hallway to this 25 feet leads to a desired 30 to 35 foot core-to-perimeter dimension. Lobbies, skylobbies, commercial spaces and mechanical floors are more adaptable to varying dimensions. Floors devoted to parking cars are planned on a 55 foot dimension so that two cars can be parked lengthwise and still leave two lanes for traffic. Thus, the floor plan's shorter dimension is constrained by its intended use and the site dimensions.

The longer dimension is more adaptable because, for example, many apartments can be laid out side by side. However, too large a long dimension in the floor plan can leave "dead" or unrentable areas beside the long dimension of the service core. Additionally, speculative office buildings often are marketed for a specific type of clientele who prefer a certain range of square footage per floor for office efficiency. Also, when constructing a building higher becomes more expensive than building it wider or in two towers, an increased floor area may be more desirable. Therefore, the floor plan dimensions are based on somewhat contradictory criteria of limited core-to-perimeter dimensions, limited site dimensions, a standard floor area for specific uses and the expense of building height. Tradeoffs between these criteria are provided by a small set of productions which alter the horizontal dimensions of the previously cubic occupiable units. Furthermore, this is the first instance in the design process where non-spatial information (occupancy class) has an impact on the geometric variables (plan dimensions) in the design.

5.2.2 Inter-volume transformations.

The previous design tasks generate a set of solids representing the necessary occupiable units according to their own architectural and structural requirements. These individual solids representing the occupiable units are now transformed to enforce inter-volume constraints and to form a cohesive massing of the entire building. Certain constraints are evident from the existing buildings designed by Khan. For example, adjacent occupiable units must be either geometrically continuous or mechanically reinforced to ensure structural continuity[4]. The set of productions that geometrically enforces inter-volume compatibility selects one or more crucial occupiable units and adapts the dimensions of the other occupiable units to the selected units. One production matches on an occupiable unit with a label whose attribute field occupancy-class has the value apartments, and then changes the horizontal dimensions of all other occupiable units to the dimensions of the apartment volume. The volumes of all occupiable units are maintained by adjusting their vertical dimension while moving the occupiable units up or down to maintain the units' non-intersecting adjacency. Another production matches on two occupiable units: a lower unit with a label whose attribute field occupancy-class has the value offices and an upper unit with a label whose attribute field occupancy-class has the value apartments. The production shapes all occupiable units into truncated pyramids, retaining the bottom face dimensions of the offices and apartments units. Then, the dimensions of each unit is uniquely determined by the dimensions of the offices and apartments units and the constraint of retaining the previous volume of every unit.

53 Instantiating a service core.

Each occupiable unit requires a service core to accommodate elevators, fire stairs, mechanical equipment and other essential, but unrented, spaces. The number and type of elevators is a result of the occupancy class and floor area of the particular occupiable units the core services. Therefore, the core dimensions of each occupiable unit are computed individually, expressed in the productions as a percentage of that occupiable unit's floor dimensions, with the particular percentage based on the unit's occupancy class. Then, the core of each occupiable unit is made compatible with the core above it. We place all cores in the center of the occupiable units because of the prevalence of this arrangement in tall buildings. During the subsequent knowledge acquisition study this assumption will be tested with practicing designers[5]. The productions which instantiate and shape the service cores work from the topmost occupiable unit downward so that (except for the topmost core) a service core volume in every unit may be placed directly below the core volume in the unit directly above it and can be no smaller than the service core above.

---

[4] In the John Hancock Center the occupiable unit devoted to offices requires a large core-to-perimeter distance whereas the occupiable unit devoted to apartments requires this distance to be smaller. The inter-volume constraints are satisfied by a geometric transformation of occupiable units into truncated four-sided pyramids with the adjacent occupiable units having geometrically continuous faces. In the Sears Building, in contrast, the requirements of the lower units call for a large square footage per floor, whereas the intended occupants of the. upper floors need a smaller square footage per floor. The differing square footage requirements are satisfied by large setbacks in the building's volume achieved by terminating individual tubes making up the bundled tube structure. The top floor of an occupiable unit is reinforced with full story trusses across the entire width of the bundled tube structure so that the deflection of the continued tubes does not result in overly large stresses where the continued tubes join the rest of the structure.

[5] A more complete grammar could include paired service cores or arrangements of larger numbers of cores, and cores with more complex shapes than the rectangular cores used here. Khan's Sears Tower, for example, uses a cruciform core plan allowing arms of the cross to be discontinued as the cells of the bundled tube are discontinued. This prototype has assumed some simplifications of the domain in order to concentrate on more crucial points.

5.4 Instantiating the abstract tube.

The structural tube is based on the idea of moving all structural members to the perimeter of the building so that they have the largest moment arm possible to optimize their resistance to bending, and tying the perimeter members together to emulate the bearing wall concept. These notions are realized, or approximated, using a small number of different system types and materials. Therefore, to serve as a spatial abstraction for subsequent refinement and as a repository for non-spatial attributes, a unit thick solid is wrapped around the exterior vertical faces of the architectural volume.

5.5 Refining the abstract tube.

The refinement of the abstract tube is performed in three stages: (1) material selection; (2) abstract member instantiation; and (3) member dimensioning. The first operation is a discrete choice, i.e., the selection of concrete or structural steel for the construction material. The abstract member instantiation is a stepwise addition of unit thick members to the design state, composing a class of tube system such as framed tube, doubly braced megaframe, etc. After the system is composed, the members can be dimensioned to meet their structural loads based on a preliminary analysis. Framed tube members (and secondary members of megaframes and trussed tubes) are instantiated and dimensioned only at four height levels in order to speed up the solids modeling and display routines. The members of the framed tube are assumed to be of constant dimensions from one level upwards until the next level is reached; the beams and columns of each floor not instantiated are assumed to have the same cross-sectional dimensions as the members on the nearest level below them.

5.5.1 Material selection.

Structural tubes have been constructed in reinforced concrete, structural steel and composite construction combining the two materials. The steel used in composite construction is employed for its constructional efficiency, and the resulting composite structure may be analyzed as though it is a normal reinforced concrete structure. Therefore, we offer two productions for adding a label to the abstract tube, a label whose attribute field `material` can take the values of `reinforced-concrete` or `structural-steel`. The value of this label is used in the subsequent stages of member dimensioning and structural analysis.

5.5.2 T\ibe class selection via abstract member instantiation.

Structural systems approaching the economy and rigidity of a cantilever structure have been classified by Khan [Khan 72, Khan 74], described in Section 2.2 and have become pan of the repertoire of practicing structural designers. The classes of structural tubes are evolving as new buildingneeds arise, but the grammar aims at characterizing Dr. Khan's building style, and therefore we use his classification as a branching point in the design space.

The various classes of structural tubes are developed within the abstract member instantiation subtask. The alternatives follow the decision tree shown in Figure 3. Decisions are shown in oval nodes and resulting systems are shown in boxed nodes. The first possible transformation divides each face of the abstract tube horizontally[6] into two or more panels so that the abstract tube can be refined into a megaframe or diagonalized framed tube. The height of the divisions are based on keeping the panel's diagonal at an angle

[6] A more complete grammar might first divide the abstract tube vertically to allow the refinement into a bundled tube.

between 35 and 55 degrees or keeping the panels as close to square as possible. Next, megaframe abstract members may be introduced, framing the panels resulting from the previous production. If the previous production is not applied, then the megaframe member instantiating production will not match on the design state. If megaframe members are instantiated, other productions can add one or two diagonal members, again of unit cross sectional dimensions, to each of the panels. If the abstract tube is not divided into panels, members of a framed tube may be introduced into the design by another production.

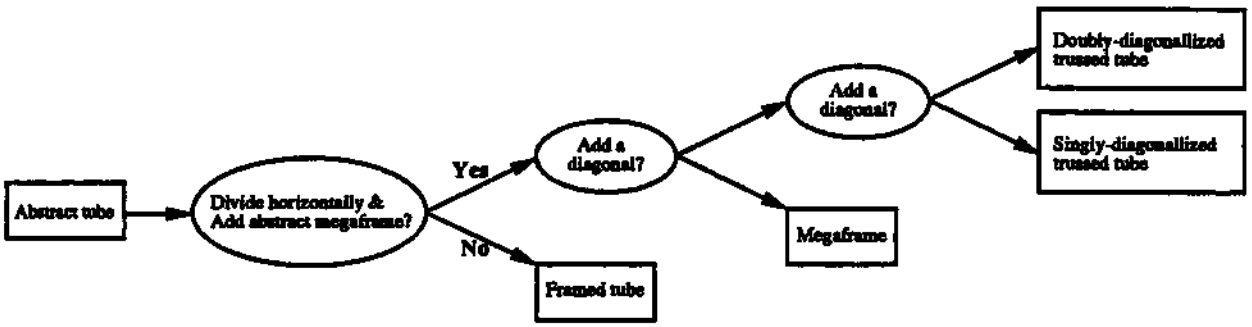


Figure 3: Selection of structural tube type by abstract member selection.

5.5.3 Member dimensioning.

Once the type of structural tube is instantiated by generating an abstract member configuration, the member forces may be determined from a preliminary analysis without regard to member cross-sectional dimensions. Assumed loads are given by the ANSI specifications of live and dead loads based on occupancy categories, and ANSI procedures for computing wind loads as a function of height based on gross geographic location. The worst-case combination of factored loads is used to compute the preliminary forces from which the cross sectional dimensions of the members are determined. The forces due to lateral and gravity loads are calculated at the four heights where the members are allowed to change cross-sectional dimensions using a modified cantilever method [Khan 73].

5.6 Instantiating secondary structural members.

The use of certain structural tube systems and the combination of certain subsystems requires the introduction of secondary structural subsystems or members. For example, when a diagonalized tube such as the one employed in Chicago's John Hancock Center is used, secondary beams and columns must be added to the structural system for handling gravity loads between the widely spaced main columns and diagonals. This subtask could also be used to add link beams that tie a structural core to the perimeter tube.

5.7 Analysis and feedback.

The preliminary structural analysis used for initially sizing member is merely a rough estimation of the actual member forces because it is based on a number of simplifying assumptions including a particular distribution of forces and member sizes. The cantilever method, for example, assumes a fixed, mathematically defined (linear) distribution of axial forces in the columns of a frame and the columns are assumed to be of equal

cross sectional dimension. However, we know from the discussion of shear lag in Section 2.2 that the distribution of axial forces is not proportional to distance from the neutral axis. The distribution of axial forces is actually nonlinear but it is unclear what this distribution actually is. Also, since the comer column is the most effective column in resisting cantilever bending it is typically larger than the other columns. Therefore, we provide a set of productions to facilitate running a two- or three-dimensional frame analysis and to graphically present the results so that the designer can get a more accurate idea of the member forces during preliminary member dimensioning. When any members are sized or resized a label[7] whose attribute is analysis-state is given the value unanalyzed When this label has the value unanalyzed the analysis productions may be applied. After the analysis, the label's value is changed to analyzed.

A production asks the designer if he or she would like to run a two-dimensional ANSYS analysis and informs them that this operation will take more time than the other productions[8]. If the designer applies this production the spatial model is queried for building dimensions, member spacings and dimensions, and loads are recalculated. This information is written to a file as a small configuration description. A shell script is waiting in the background of the host machine that sends this configuration file to the remote machine that can run the analysis[9].

The remote machine also has a shell script running in the background looking for the existence of the configuration file. When the configuration file arrives, the shell script first must turn the configuration file into an analysis input file. The shell script runs an awk script to write the ANSYS input file, runs the ANSYS analysis and then runs another awk script on the ANSYS output to extract a relatively small set of representative member forces to return to the host machine as a file containing a single nested list of member forces. Then the configuration file is renamed as a backup so that the shell script can wait for a call for another analysis.

When the file of computed member forces is returned to the host machine the shell script waiting there ends the application of the analysis production. The next applicable production, now that a new analysis file resides on the host machine, graphically displays the recently computed member forces along with the assumed forces previously used to size the members. We extract two sets of member forces from the ANSYS output: axial forces in the columns and shear forces in the spandrel beams at each of the four levels where members arc instantiated. If the designer applies the analysis display production, two pairs of forces are displayed: assumed and computed axial forces, and assumed and computed shear forces. By overlaying the graphic display of the two sets of forces, the assumed forces can be visually compared with the computed forces. The forces are displayed by introducing vertical laminae at column and beam locations around two sides of the building. The lamina's height is scaled to the force it is displaying and the four forces are differentiated by using different colors for the lamina, and by pairing the axial forces above the level and the shears below the level.

5.8 Modification of members in response to analysis.

When the frame analysis results are displayed and compared with the assumed forces, it is likely that the assumed and computed forces will not correspond exactly. In this case, the members may be inadequate for

---

[7] **This label is arbitrarily attached to a ground floor corner column.**

[8] **A typical production simply altering a few shapes or labels may take from 1-5 seconds. In contrast a 2D ANS Y analysis of a 45 story framed tube can take 4-10 minutes and a 3D analysis can take 30-45 minutes on a Sun4.**

[9] **Of course, this would not be necessary if the host machine could also run the analysis package, but that is not the case for us.**

resisting the computed forces, or the members may be uneconomically overbuilt. Three productions allow the resizing of three classes of members: resizing the coiner columns, resizing the mid-face columns and resizing the spandrel beams. Each of these productions asks the designer for a list of scaling factors to be applied to a particular class of members. The listed scaling factors allow the designer to scale members at different levels by different factors. This production also changes the value of the analysis-state label to unanalyzed so that the designer may request another analysis. Alternately, the designer may proceed to the next subtask rather than perform another analysis.

5.9 Detailing of tube system, e.g. ground floor modifications.

The configuration of the tube at this point has made no provisions for local requirements such as entryways. The architectural expression of the buildinghas focused on large scale features such as volumes and setbacks. Further architectural detailing is not handled by the prototype grammar, but structural modifications to accommodate entryways and other small features are provided by a few productions. One production removes every other column on the ground floor. The application of this production requires that the spandrels above the ground floor also be modified. Therefore a second production resizes the spandrels above the ground floor as a transfer beam.

5.10 Looping when analysis evaluation warrants.

The local modification of the tube, performed in the previous subtask, is accomplished by a production that changes the value field of the analysis-state label to unanalyzed since the previously analyzed model is no longer representative of the current design state. Thus, the designer is asked if he or she wishes to perform another analysis, again being given the choice between a two- or three-dimensional frame analysis. In the prototype we did not incorporate any exit requirements for this loop. The designer may perform an analysis and then, even without evaluating the analysis results, he or she may terminate the design process.

5.11 Summary of Design Process

It can be seen from the above description of the design process that a hierarchical process is developed in terms of solid object generation as well as during object parameterization. During generation, the occupiable units begin as a single solid and are then transformed into an instance-specific number of solids, and the structural tube begins as a solid sheath resembling an unpierced bearing wall that is successively transformed into individual members of framed or trussed walls. The dimensioning of the solids is also hierarchical in that the objects begin as unit dimensioned solids and are then given proportions based on a series of additional constraints. The occupiable units begin as unit cubes and are then repeatedly transformed according to volumetric, core-to-perimeter, and then compatibility constraints. The structural members begin as unit cross-sectioned members, and are then given dimensions based on a preliminary analysis before being scaled by the designer based on his or her evaluation of the frame analysis results.

The attributes which express functional constraints on the design are also inserted into the design state as labels on solid objects and then later productions can modify the label's value. The modification of label values is performed less frequently than the modification of solids, but altering label values is nevertheless crucial to recursive or cyclic operations such as the sizing members first based on an approximate analysis by the cantilever method and then by a more precise frame analysis using ANSYS. The redisign of the

concrete framed tube's members can be accomplished, in addition to changing the member dimensions, by changing the construction material's properties to increase or decrease the material strength or reinforcement ratio. Labels are also used to flag consistency in the design state, such as the correspondence between the current dimensions of structural members and the currently stored analysis results via the value of the analysis-state label. Thus, both solids and labels are introduced into the design state with default geometry/values when they become a necessary descriptor of the design stage, and are then modified to have more specific geometry/values based on the context provided by the design state.

Additionally, it can be seen that spatial and non-spatial aspects of the design are tightly coupled in many of the transfoimations within the design process, just as many of the architectural and stnicturai requirements are interwoven in many of the productions. The dimensions of the occupiable units, for example, are a spatial aspect of the design which is based on a tradeoff between constraints imposed by the site dimensions and the floor area requirements, but also by the occupancy class—a non-spatial attribute of the occupiable units. Furthermore, the enforcement of compatibility between the occupiable units has some architectural basis—large overhangs would cause deep shadows in the lower occupiable unit—but it is more of a structural motivation that requires the occupiable units to be geometrically or mechanically compatible.

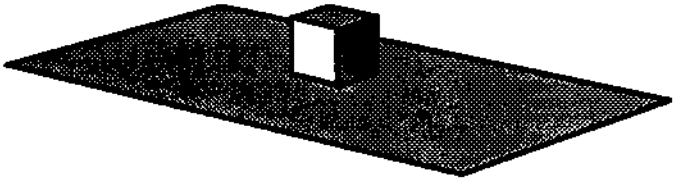
In spite of the fonnalization of the design process, the search for a path through the design space is still highly underconstrained. The production's presentation of information about the design state and the ability to query the design state add more information to the graphic presentation, helping the designer make informed decisions while guiding the traversal of the space.

6 Design Example, DeWitt Chestnut Apartments

The DeWitt Chestnut Apartments, built in Chicago in 1963, was Khan's first tube structure. This concrete framed tube is 43 stories tall and has a floor plan approximately 121 feet by 77 feet, providing a total square footage of approximately 400,000 square feet The building contains a single-floor lobby and 42 stories of apartments. In this example we have added the requirement of 40,000 square feet of commercial space to make the problem slightly more complex. In the following subsections we describe the design states by pictorially showing the solid model and listing the labels attached to the spatial elements of the model. The objea names (the first field of the triplet) are fabricated in order to clarify to which object the labels are attached.

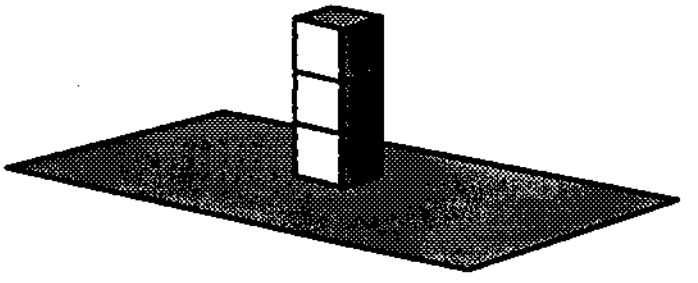
6.1 State after establishing the initial conditions

(lamina, site-dimensions, [150,90]),
(lamina, ansi-location, [90, a, 1500]),
(occupiable-unitl, occupancy-classes,[lobby, commercial, apartments]),
(occupiable-unitl, floor-areas, [10000,40000,400000]).



6.2 State after generating architectural volumes

(lamina, site-dimensions, [150,90]),
(lamina, ansi-location, [90, a, 1500]),
(occupiable-unitl, index, 1),
(occupiable-unitl, occupancy-class, lobby),
(occupiable-unitl, floor-area, 10000),
(occupiable-unitl, floor-to-floor-ht, 16),
(occupiable-unitl, ideal-lease-span, 45),
(occupiable-unitl, index, 2),
(occupiable-unitl, occupancy-class, commercial),
(occupiable-unit2, floor-area, 40000),
(occupiable-unit2, floor-to-floor-ht, 14),
(occupiable-unit2, ideal-lease-span, 40),
(occupiable-unit3, index, 3),
(occupiable-unit3, occupancy-class, apartments),
(occupiable-unit3, floor-area, 400000),
(occupiable-unit3, floor-to-floor-ht, 11),
(occupiable-unit3, ideal-lease-span, 30).



6.3 State after architectural intra-volume transformations

labels same as above.

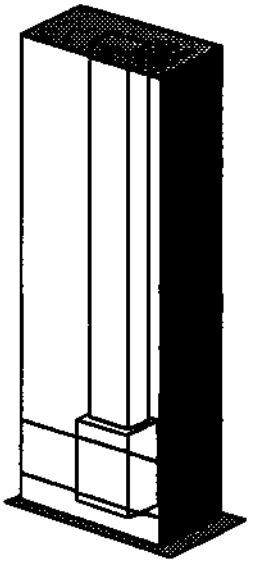
6.4 State after architectural inter-volume transformations

labels same as above.

6.5 State after instantiating a service core

labels same as above plus:

(core-unitl, core-index, 1),
(core-unit2, core-index, 2),
(core-unit3, core-index, 3).

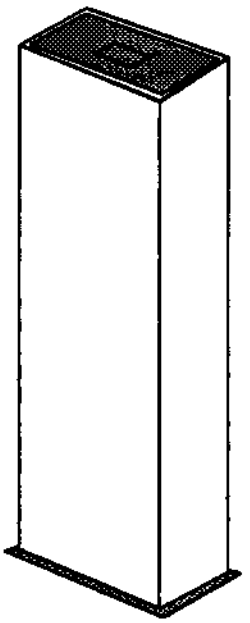


6.6 State after instantiating the abstract tube

labels same as above plus:

(tube, interior-lateral-support, nil).

6.7 State after refining the abstract tube: Material selection



labels same as above plus:

(tube, material, reinfbrced-concrete),
(tube, concrete-strength-psi, 6000),
(tube, concrete-density-pcf, 150),
(tube, rho-g, 0.05),
(tube, rebar-strength-psi, 75000),
(tube, concrete-bm-db-ratio, 25),
(tube, concrete-col-db-ratio, 05).

6.8 State after refining the abstract tube: Tube class selection via abstract member instantiation

labels same as above.

6.9 State after refining the abstract tube: Member dimensioning

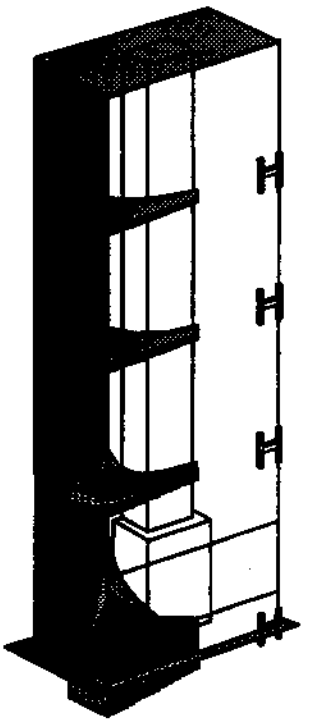
labels same as above plus:

(corner-columnl, subassembly-index, 1),
(corner-columnl, corner-axial-force-ldps, 1986.39),
(corner-columnl, stiffness-factor, 229068),
(corner-columnl, wind-psf-ht, 2.4832),
(corner-column2, subassembly-index, 2),
(corner-columnl, corner-axial-force-kips, 1544.74),
(corner-columnl, stiffness-factor, 0.71875),
(corner-column2, wind-psf-ht, 10.4772),
(corner-column3, subassembly-index, 3),
(corner-column3, corner-axial-force-kips, 105925),
(corner-column3, stiffness-factor, 0.71875),
(corner-column3, wind-psf-ht, 16.482),
(corner-column4, subassembly-index, 4),
(corner-colurnn4, corner-axial-force-kips, 529.625),
(corner-column4, stiffness-factor, 0.646076),
(corner-column4, wind-psf-ht, 21.9608),
(tube, wind-psf-top, 25.1039),
(tube, analysis-state, unanalyzed).

6.10 State after analysis and feedback

labels same as above plus:

(tube, analysis-state, analyzed),
(tube, graphic-norm, 0.0064155),
(assumed-force-laminae, analysis-iter, previous),
(computed-force-laminae, analysis-iter, current).



6.11 State after modifying members in response to analysis

labels removed:

(tube, analysis-state, analyzed).

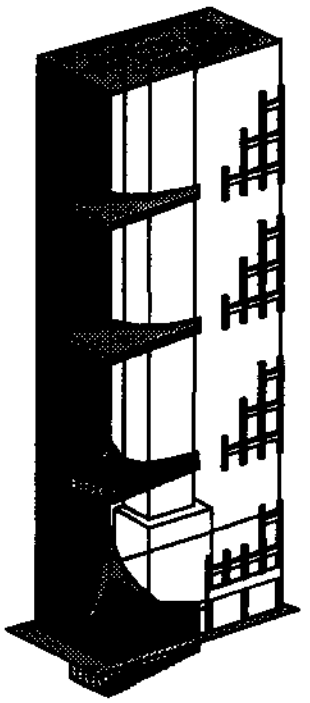
labels added:

(tube, analysis-state, unanalyzed)

6.12 State after detailing the tube system

labels same as above plus:

(corner-column!, ground-floor, modified)



7 Discussion

In the previous sections we have presented the prototype grammar by first describing our motivation and the portions of the problem we are focusing on. Then we described how we address these problems by presenting the representation, design process and a resulting design. The prototype's success can be measured by how well the problems actually are addressed and by what is learned from this experiment. Section 1.2 laid out the project's goals which are discussed in the following subsections:

1. Learning the relevant attributes of **the** domain of tall building design.
2. Formulating high-level integrated spatial and non-spatial operations for design generation.
3. Implementing and evaluating an attributed spatial grammar.

In the following sections we discuss each of these tasks in turn.

7.1 Relevant Attributes

The approach to uncovering the relevant domain attributes is to organize the design process into the tasks described in Section 5 and then to develop the operations needed by each of these design tasks. Within these operations, the necessary non-spatial information must be carried by labels attached to the solids model portion of the design state. By correlating and reducing the labels used in the operations, a minimal set of functional attributes is uncovered. The set of attributes sufficient for describing a design state, as might be done by the blueprints and specifications of a finished design, proves to be a smaller set than the attributes needed for arriving at this state. Therefore, the uncovering of relevant attributes is a subtask of learning the relevant operations within the design process.

In uncovering the crucial spatial and non-spatial aspects of the domain for descriptive and operational purposes at this first approximation, the set of required attributes proves to be relatively small. This is advantageous in that the specification of an extremely large set of attributes and their transformations is unwieldy and impractical. However, the semantics of the spatial transformations, particularly the architectural transformations, have a much broader and less rational basis than, for example, the relatively straightforward sizing of a column to resist axial forces. The overriding architectural concerns—concerns that form the semantic context within which operations such as column sizing must fit—have their basis in a very subjective open set of sociological imperatives such as what constitutes a sufficient and desirable external view, or what are appropriate dimensions for a work or living space. Thus, the aspects of a design state that form the condition of a transformation have their basis in sociological, constructional, economic and legal concerns. As such, it is possible to arrive at **a** set of attributes for performing an operation, but it is difficult to say if it is **the** set of attributes that would be used in practice[10].

The attributes necessary for the functional description and transformation of the partial design may be divided into three categories of: (1) occupancy; (2) material; and (3) structural function (i.e., applied loads or member forces.) Occupancy attributes include the intended use of an occupiable solid such as an office

[10] As a prototype, the grammar is intended to be a depth-first coverage of the domain. Many of the design tasks are not covered in sufficient breath to mirror the complete range of relevant systems or the range of conditions that constrain their applicability. Nevertheless, a first approximation of the controlling attributes are found to be expressible by the coupled solids model and object-attribute-value structures. The attributes used in practice will be investigated further during the knowledge acquisition study that forms another part of this project

or apartment volume, and also such attributes as required floor area and preliminary floor-to-floor height. These attributes must be part of the initial state so that the parametric productions can generate a specific building. The material attributes include construction materials properties used to select the structural members as well as in the selection of member dimensioning pnxiuctions. Applied load attributes are used to proportion and dimension members of the structural system and to find member forces which are used to describe the behavior of the tentatively sized members so that the designer can evaluate the adequacy of the structural system as a whole. Together these three classes of attributes express the functionality of the building, constraining the language of all conceivable tall building designs to a specific subset of the language appropriate to the needs of a particular site and usage, and producing dimensionally defined designs from a parametrically specified grammar. Only the types of labels whose values are drawn from non-overiapping enumerated sets (e.g., occupancy class attribute and material attribute of concrete versus steel) could be unambiguously expressed using the shape grammar labeling formalism of object-value labels because the labels value would then clarify its own metric context. Any numeric valued labels, or label's whose values are drawn from contrasting contexts, arc ambiguous without a clarifying attribute field.

7.2 High-Level Operations

We define high-level operations as those predicates or productions that transform the model without the designer having to directly manipulate the solids model or the label database. That is, high-level operations divorce the designer from having to employ Eulcr operations, geometric transformation matrices or manipulate the model directly to perform a design operation, querying or transforming an element of the design state. High-level operations also maintain the integrity of the model by adjusting elements of the model when the focal elements of an operation are transformed. The value of high-level spatial operations, and integrated spatial and non-spatial operations, is based on their usefulness over a broad range of applications. Many operations in the prototype have been built hierarchically from smaller operations, and it is more frequent that these smaller operations are used over and over again, rather than the higher level operations being used in many design operations. For example, during the resizing of occupiable units and service cores the vertical translation of occupiable units and cores above the volume being resized is performed by a simple predicate as part of the resizing operation. Likewise, the transformation of the occupiable units into a pyramidal volume uses the same geometric transformations as the later transformation of the megaframe, initially instantiated with vertical columns, into a pyramidal configuration. Similarly, queries about floor areas, number of floors, adjacency of occupancy types, are all built as simple predicates which are used repeatedly by higher-level transformation and query predicates.

This latter class of high-level operations—queries—consists of the presentation of information which may be valuable to the designer in making decisions about applying or not applying a particular rule to the current design state. Information that is certainly pertinent is displayed automatically when the suggested rule is found to unify with the current state. The textual description of the rule is always presented, and any specific topological element or solid referred to by the rule description is highlighted in the display of the solid model. This highlighting of topological elements that unify with crucial clauses of the production is particularly important when a rule can apply to one of many similar elements or solids in the design state. Additional information, particularly parametric information such as the dimensions or capacity of portions of the solids model, is textually displayed to the designer before he or she is asked to decide to apply the rule or not. A third method of presenting pertinent information through the availability of high-level operations is the ability to directly query the model and label database.

Many of these high-level operations stretch the boundary of the shape grammar definition. For example, the use of a frame analysis program to introduce and dimension four series of laminae describing assumed and computed member forces is a transformation of the existing model by a domain specific integrated geometric and non-geometric transformation rather than a straightforward rewriting of an existing element of the model. The basic mechanism of the dimensioning of these laminas is through a series of matrix manipulations, but these matrices are more complex and domain-dependent than those used in the standard geometric transformations of translation, rotation, scaling, etc. Similarly, the productions that ask the designer for a list of scaling factors to modify the dimensions of structural members involves the designer as an essential ingredient in the definition of the design space.

73 Implementation

A parametric definition of the axiom state and the productions facilitates the parametric definition of a design language that is a response to instance-specific functional requirements. In the specification of the grammar, the initial state is incomplete without the addition of instance-specific attributes such as the site dimensions, number and occupancy class of the occupiable units, and geographic location. These attributes are used extensively in productions that transform the number and dimensions of the occupiable units and that select and size the structural systems. The parameterization of productions may be thought of as being accomplished in two different ways: topological parameterization and geometric parameterization. In the first case, productions are written with topological variables to allow the generalization of operations to multi-solid operations; to an unspecified number of topological elements; or to solids that satisfy a set of topological constraints. In the second case, productions are written using geometric variables that parameterize topologically consistent operations which describe principles of spatial organization.

In the prototype grammar, both in the specification of the initial labeled solid and in the productions themselves, we adopt the approach of setting the attribute field of a label and allowing the object and value fields to be variables. A typical clause that matches on a label with, for example a material attribute field, label(S, material, V), can be paraphrased as 'Find a label with the attribute field material and assign the variable S the topological object with that label and the variable V the value field of the label.[9] Clauses in the condition of this production can add constraints to discount matching on this design state if the consequent of the production should be performed on the object S only when it is of material V so that the consequent can then perform operations that are appropriate to the material V.

This example shows how the object-attribute-value structure can express labels whose value field are from an enumerated set by matching on the label's attribute field. The parameterization of label values is also necessary when describing functional attributes that can take continuous values, and for describing operations that perform transformations that are mathematical functions of contextual labels. For example, the generation of the vertical sequence of occupiable units is the result of a production that generates a variable number of unit cubes with a defined number of labels whose attribute field is fixed, but whose value field is variable. This sequence of solids is generated by the repeated application of a predicate that iterates over the occupancy class list. This occupancy class list is a label on an initial solid that gets its values, and therefore its length, from the designer. The values in the initial list are distributed to the unit cubes as they are generated along with the other labels describing functional attributes such as required square footage and preliminary floor-to-floor height. Then, the volumetric transformations of the occupiable units are expressed as mathematical functions of the discrete and continuous values of the labels attached to each solid. When expressing labels with variable values, matching on the attribute field rather than on a geometric

location allows the label's object field to be a variable denoting a specific topological object, and allows the value field to use a broader range of data types thereby allowing the value to be used in a broader range of subsequent computations. We thereby have a more expressive vocabulary that leads to a larger repertoire of possible operations.

What additional subprocesses should be integrated in the design process? The integration of structural analysis with the instantiation and (re)sizing of structural tube members is based on a timely and graphic presentation of the analysis results within the same environment as the generation and modification of the tube system. The assurance of structural adequacy is certainly a necessity in generating the language of designs since all Dr. Khan's tube structure designs, either built or unbuilt, had their structural fitness as a necessary condition. However, it is unclear whether we should require a structural analysis as part of the design process during the preliminary design stage. Therefore, we offer this subtask to the designer, but don't require its use. Additional subprocesses within the design generation process could include an evaluation of the comparative construction cost and construction time, but these factors are not germane to the inclusion or exclusion of a design from the language. It could be argued that the construction cost and time are pertinent attributes of a design that are included with its physical description during preliminary design. A more important design subtask that is not addressed within the prototype grammar is a more detailed modification and elaboration of the floor plan, including the layout of the service core. The floor plan of a tall building is still the central organizational spatial view since the tall building is based on the large scale repetition of a few sequentially related floor plans. The aim of elucidating organizational principles of the domain warrants developing the architectural volumes starting from floor plan development and then projecting upward. The service core is a crucial element of these floor plans, both as individual floors and as an element that relates the plans as the perimeter shape changes. As unrentablc space, the area allotted to the service core should be kept to a minimum. By not including a design subtask that details the service core layout and attempts to minimize its floor space, we may have excluded attributes and operations that would have affected our appraisal of the prototype.

How expressive and convenient is a solids modeling representation for the specification if a design language? At the more detailed levels the solids model provided a complete representation of the topological and geometric aspects of the design state. However, it is often too cumbersome having to specify topological elements by following edge-halves and face loops. This is especially true at high levels of abstraction when manipulating solids that represented abstract systems such as the unit thick abstract tube.

7.4 Evaluation of Syntactic Richness

The parametric definition of labels and productions allows the compositional definition of a specific subset of the total design language that the grammar can generate. This self-constraining of the grammar restricts the generated designs to a subclass which meets some rough functional requirements such as buildings that contain a lobby and apartment volumes and a desired total volume. When the designer is searching through the design space he or she should not have to search through designs that respond to different functional requirements. Through responding to instance-specific functional requirements expressed as the values of variable labels the design language is constrained to a *semaniicalty* constrained subset of the design space.

The specification of a design language through the definition of a state representation and closed operations on that representation is a central theme of the grammar description of a domain. The use of the grammar model for describing an engineering domain bounded by functional constraints has proven difficult in the past due to the lack of a rich representation for non-spatial aspects of the design state, as

well as the poor integration of non-spatial and spatial aspects of the design. These issues are addressed with mixed results in this prototype. The representation for spatial aspects is well formed, that is, it is closed and complete. Unfortunately, the same still cannot be said for the non-spatial aspects. We cannot say that just because a label's value is transformed from one string to another string that it is a closed representation. It may be syntactically closed, but not semantically closed. Thus, the parameterization of label values increases the formal problem of ensuring the semantic validity of the state representation with respect to the domain. Some formal specification of the label transformations and their semantic basis must also be provided to relate the productions to the semantics of the domain they are modeling. The use of an algebra is appropriate for a purely syntactic representation such as geometric and solids models, but for semantically controlled aspects such as the non-spatial attributes of an engineering domain, the specification languages that procedurally define the transformations appears more appropriate for capturing the mechanism of the operations in addition to defining what the types and ranges of the input and output.

The number and diversity of designs producible by the prototype grammar is admittedly small. Likewise, the deductions about the organization of the domain must be considered as a preliminary hypothesis to be tested by a more extensive grammar. Nevertheless, the expressive quality of the combined labeled solid representation using and object-attribute-value structure for labels can be considered an evolutionary improvement of the simpler object-value labeled shape grammar formalism. Discounting slight geometric differences in the designs resulting from the parametric grammar, the classes of designs can first be divided into three classes along the method of compatibility enforcement: sequences of occupiable units can be made compatible by geometric smoothing into a vertical sided prism; by geometric smoothing into a vertically tapering prism; or by using symmetrical setbacks. The division along construction material may be important, but that importance is not brought out by the prototype. For example, the use of a framed tube has been called "more natural" in concrete due to the material's monolithic nature, avoiding the numerous connections of steel. However, the practical choice of concrete over steel is for the designer to make, and there have been roughly as many framed tubes designed in concrete as in steel. Similarly, the grammar makes no judgment about the use of concrete in a diagonalized tube because there is such a structure in the corpus of Dr. Khan. The designs in the language generated by the prototype grammar can be divided along the type of tube structure instantiated. The language includes framed tubes, singly and doubly diagonalized tubes and megaframe tubes. Thus, we have a small number of design classes in the language $L(G)$ where the cardinality of the classes $C(L(G)) = 8$ because diagonalized tubes are not compatible with setback volumes.

Despite the shortcomings of the prototype, it has demonstrated the utility of labeled spatial grammars as a description of a design domain, i.e. as a knowledge representation able to integrate spatial and non-spatial data. The solids model and label structures representing a design state are an unambiguous descriptive expression frozen in time, whereas the rules of the grammar are a parametric prescriptive expression of the operations in the domain, applicable to a range of states described by the rule's condition. The integration of a descriptive expression of the state and a prescriptive definition of transformations provides a representation appropriate to the needs of the design process.

8 Conclusions

The presentation of shape grammars typically demonstrates the generation process by pictorially showing the productions and tracing the path of one design. In presenting the productions in the grammar, any labels are also presented in the two-dimensional pictorial representation of the productions, and any constraints on the labels are given as short equations or inequalities. These grammars have typically focused on generating

the floor plans of buildings such as Palladian villas and Wright's prairie houses. The demonstration of the organizational structure of these classes of three-dimensional artifacts is based on the grammar's exposition of the structure of their two-dimensional floor plans. We have not shown any productions in this description of our grammar because of the difficulty of pictorially presenting a production's three-dimensional parametric character along with the extensive set of mathematical and logical constraints within each production. This is not only a drawback in a paper such as this, but more so it is a difficulty during the development, debugging and in-person demonstrations of the grammar. One supposed advantage of production systems is their ability to present the captured domain knowledge in a way that it may be inspected by the designer. The functionally descriptive labels and their transformation by the predicates of the productions should procedurally describe why and how the parameters are derived that way, not just to say what the result of the transformation is. However, the complexity of the parametric three-dimensional transformations and the constraints on contextual conditions and label value derivations makes it difficult to pictorially describe a production except by a contextual example that deemphasizes its parametric generality. A corollary to this complexity is that when deciding on the granularity of a production, the topological and geometric complexity, in contrast to its conceptual simplicity, leads to a balancing act between expressiveness and generality on one hand and presentability and simplicity on the other.

What have we learned about the formalism? The integration of spatial and non-spatial data expands the design process to search previously unattainable points in the design space, while parameterization of the axiom state and the productions constrains the search space leading to a more efficient search, or, design process. A solids model representation augmented with object-attribute-value structures is adequate for representing pertinent information, but the representation may be too low-level, and therefore cumbersome for preliminary stages of the design process. Thus, a better mechanism is needed to express hierarchical and other organizations of the design artifact and design process.

What have we learned about the domain? This prototype is more an exercise in extending the representational formalism of labeled solids grammars than an exposition of the organizational structure of tall building design. Only a shallow domain structure is demonstrated by the prototype. The architectural style of the tall buildings of Fazlur Khan is founded on a logic of function. It is the functional logic of the International Style based on the utility of tall buildings as places for people to work and live. Most of these buildings are based on a rectangular plan with numerous very similar floors. The few buildings with non-rectangular plans or elevations do so in response to the complexity of the site (e.g., One Magnificent Mile) or to the differing functional requirements of the tenants (e.g., Onterie Center, and the Sears Tower). The prototype did not get to the stage of development where it could model these types of buildings, but the functional basis for such a subclass is developed, as demonstrated by modeling the smoothly non-rectangular elevation of the John Hancock center in Chicago. The extension of the logical basis of the prototype grammar to more recent buildings employing a tube structural system would be difficult when the architectural volumes of these buildings are developed on a less rational basis and on less geometrically regular plans and elevations.

It is a paradox that the freedom of developing a design process within the grammar paradigm can be a strength and a weakness of the paradigm due to the lack of guidance or assistance in managing the design process. There is no inherent design method dictated by the grammar model, only domain-independent syntactic transformations. There is also no innate abstraction mechanism. The developer of the grammar must devise all of the mechanisms for the employed design method and abstractions from the same vocabulary used to represent the design state. The implicit generate-and-test strategy of the model consists of recursively transforming a state until no rules will unify with it; then if there are still non-terminal elements in the state the design is invalid and disregarded. There are no repairing modifications possible through the application

of rules since no more rules are applicable to the state; the partial design is simply thrown away. There is evidence that practical design involves the continuous testing and modification of partial designs. A design model that does not allow modification of inadequate designs requires starting over when the inadequacy is noticed. A gcnerate-and-test strategy that includes modification of inadequate designs is more efficient and more reflective of design practice, allowing a more natural interaction between the designer and the machine. The mixture of hierarchical refinement, and sequential, recursive and iterative design operations employed in the prototype grammar is possible due to the grammar paradigm's lack of an imposed consistent design methodology. Perhaps this mixture, and the freedom to select diverse methodologies at various stages of the design process, is more reflective of the organization of the design process than any single method could be.

References

[ANSI 72] *American National Standard Building Code Requirements for Minimum Design Loads in Buildings and other Structures.* American Standards Institute, New York, ANSI A58.1-1972 edition, 1972.

[Baker 89] N. C. Baker. "Towards a spatial and functional building design system." PhD thesis, Carnegie Mellon University, Pittsburgh, PA, January 1989.

[Beedle 82] L. S. Beedle and S. H. Iyengar. "Selected Works of Fazlur R. Khan (1929-1982)." *International Association of Bridge and Structural Engineers, Structures,* C-23/82:63-82, November 1982.

[Coyne 89] R. COyne. "Planning in design synthesis: Abstraction-Based LOOS (ABLOOS)." Technical Report EDRC 48-14-89, Engineering Design Research Center, Carnegie Mellon University, 1989.

[Downing 81] F. Downing and U. Hemming. 'The bungalows of Buffalo." *Environment and Planning B: Planning and Design,* 8:269-293,1981.

[Fenves 76] S. Fenves and S. Liadis. "A data structure for computer-aided design of buildings." *APEC Journal,* pages 14-18,1976.

[Finger 89] S. Finger and J. Rinderie. "A transformational approach to mechanical design using a bond graph grammar." In *Proceedings, Design Theory and Methodology Conference,* ASME, Montreal, September 1989.

[Hemming 81] U. Hemming. "The secret of the Casa Giuliani Frigerio." *Environment and Planning B: Planning and Design,* 8:87-96,1981.

[Hemming 86] U. Hemming. "On the representation and generation of loosely packed arrangement of rectangles." *Environment and Planning B: Planning and Design,* 13:189-205,1986.

[Hemming 88] U. Hemming. *Rule-Based Systems in Computer-Aided Architectural Design,* pages 93-112. Academic Press Inc., N.Y.C, 1988.

[Hemming 89a] U. Hemming. "More on the representation and generation of loosely packed arrangement of rectangles." *Environment and Planning B: Planning and Design,* 16:327-359,1989.

[Hemming 89b] U. Hemming, R. Coyne, T. Glavin, H. Hsi, and M. Rychener. "A generative expert system for the design of building layouts, final report." Technical Report EDRC 48-15-89, Engineering Design Research Center, Carnegie Mellon University, 1989.

[Heissennan 90] J. Heissennan. "Generative geometric design and boundary solid grammars." Technical report, Engineering Design Research Center, Carnegie Mellon University, 1990.

[Iyengar72a] S. H. Iyengar. "Preliminary Design and Optimization of Steel Building Systems." In *Proceedings of the International Conference on Planning and Design of Tall Buildings,* Lehigh University, Bethlehem, PA, N.Y.C, American Society of Civil Engineers, August 1972.

[Iycngar72b] S. H. Iyengar, N. Amin, and L. Carpenter. "Computerized Design of World's Tallest Building." *Computers and Structures,* 2(5-6):771-783, December 1972.

[Jaffar90] J. Jaffar, S. Michaylov, P. J. Stuckey, and R. H. C. Yap. "The CLP(R) language and system." Technical Report CMU-CS-90-181, Department of Computer Science, Carnegie Mellon University; October 1990.

[Khan o7] F. R. Khan. 'The John Hancock Cfenter." *Civil Engineering,* 37:38-42, October 1967.

[Khan 72] F. R. Khan. "Influence of Design Criteria on Selection of Structural Systems for Tall Buildings." In *Canadian Structural Engineering Conference Proceedings,* Canadian Steel Industry Construction Council, Toronto, Canada, 1972.

[Khan 73] F. R. Khan. "Analysis and design of framed tube structures for tall concrete buildings." *Structural Engineer,* 51(3):85-92, March 1973.

[Khan 74] F. R. Khan. "New structural systems for tall buildings and their scale effects on cities." In *Tall Buildings, Planning, Design, and Construction, Symposium Proceedings,* pages 99-128, Nashville, Tennessee, Vanderbilt University, November 14-15 1974.

[Koning 81] H. Koning and J. Eizenberg. "The language of the prarie: Frank Lloyd Wright's prairie houses." *Environment and Planning B,* 8:295-323,1981.

[Krishnamurti 78] R. Krishnamurti and R H. O. Roe. "Algorithmic aspects of plan generation and enumeration." *Environment and Planning B: Planning and Design,* 5:157-177,1978.

[Maher 85] M. Maher and S. Fenves. "HI-RISE: A knowledge-based expert system for the preliminary structural design of high rise buildings." Technical Report R-85-146, Department of Civil Engineering, Carnegie Mellon University, January 1985.

[Mitchell 90] W. J. Mitchell, R. S. Liggett, and M. Tan. 'Top-down knowledge-based design." In M. McCullough, W. J. Mitchell, and R Purcell, editors, *The Electronic Design Studio,* pages 137-148. The MIT Press, Cambridge, MA, 1990.

[Mitchell 91] W. J. Mitchell, R. S. Liggett, S. N. Pollalis, and M. Tan. "Integrating shape grammars and design analysis." In G. N. Schmitt, editor, *CAAD Futures' 91, Proceedings: Education, Research, Applications, pages* 1-18, Zurich, Switzerland, 1991.

[Stiny75] G. Stiny. *Pictorial and Formal Aspects of Shape and Shape Grammars: On Computer Generation of Aesthetic Objects.* Biikhäuser, Basel, 1975.

[Stiny78a] G. Stiny and J. Gips. *Algorithmic Aesthetics: Computer Models for Criticism and Design in the Arts.* University of California Press, 1978.

[Stiny78b] G. Stiny and W. J. Mitchell. "The Palladian grammar." *Environment and Planning B: Planning and Design,* 5:5-18,1978.

[Stiny 80] G. Stiny. "Introduction to shape and shape grammars." *Environment and Planning B: Planning and Design,* 7:343-351,1980.