WHAT IS A DESIGN?

George Stiny

Graduate School of Architecture and Urban Planning
University of California, Los Angeles

ABSTRACT

Designs belong to relations.

Practice has it that designs are complicated and multifaceted. In the basic case, a design consists of line drawings that describe something for making and show how it works. Drawings may be used singly, or they may be used multiply, for example, to establish three-dimensional relationships in plans, sections, and elevations, or to describe the separate parts and their relationships in an assembly. Drawings, however, rarely give everything in a design. Sometimes, they are augmented with solid models and other kinds of geometrical representations; more generally, they are combined with labels and samples to describe form further and to provide details of function, material, construction, and so on. Drawings themselves may be described in different ways by decomposing them into parts that may be ordered hierarchically or in some other way, and by assigning these parts to categories to clarify intention from different points of view and to facilitate analyses of one sort or another. Drawings, too, may be identified with other descriptions such as diagrams, graphs, networks, and mathematical expressions for these purposes. And the use and scope of the various descriptions in a design may depend on special instructions and ancillary documents, or be elaborated by additional commentary. In practice, there are many reasons to connect many devices of many kinds to make a design.

The complexity of designs suggests a simple definition. A design is an element in an n-ary relation among drawings, other kinds of descriptions, and correlative devices as needed.
This is more than mere bookkeeping; it provides a way of looking at designs in terms of algebras and linked computations in these algebras that are carried out by following rules. A relation containing designs is defined recursively in an algebra that is the Cartesian product of others. The properties of the algebra depend on the properties of the algebras that combine to make it, and the properties of the relation depend on computations in these algebras and on how they influence one another as they are performed in parallel to produce designs.

Broad avenues of research open once designs are viewed in this way as elements in relations defined in algebras. The algebras themselves contain the expressive devices used by specialists in different domains of interest, say, descriptions of form, of function, or of construction, manufacture, and assembly, and have to be characterized, each in its own right, before any progress can be made. The ways in which these algebras combine allowing for specialists to communicate across domains need to be examined if designs are to involve multiple perspectives. Useful methods of computation in different algebras and schemes to link them to produce the designs in relations require investigation. And particular relations containing designs of things in certain styles and of certain types must be defined and explored in practice, and extended and changed to have continuing use as practice responds to new interests and goals.

The rudiments of these approaches are illustrated nicely in the shape grammar formalism, and in its generalizations. Shape grammars exploit two basic algebras -- the algebra of shapes $U^*$, and the algebra of sets of labeled points $V^*$ -- that are combined and elaborated in Cartesian products and with other set-theoretic techniques to define relations containing designs.

Line drawings and their counterparts in space are shapes. Shapes have an algebra $U^*$ that is equivalent in every important respect to drawing with pencil on paper. In $U^*$, shapes may be decomposed into parts in accordance with a subshape relation in any way whatsoever, and they may be manipulated with two operations -- sum and difference -- that correspond to drawing and erasing lines, and with the Euclidean transformations. The details of $U^*$ are characterized conveniently in terms of a simple relation on lines.

Every line is given by endpoints that are always distinct and has finite, nonzero length. Lines defined in this way are related according to whether one is embedded in another. This relation holds for the lines $l$ and $l'$ if $l$ is a segment of $l'$. In this case, each of the endpoints of $l$ is either an endpoint of $l'$ or falls on $l'$ between its endpoints. The embedding relation is a partial order on lines: it is transitive, and two
lines are identical whenever they are embedded in one another. This gives every line the same structure. All of the lines embedded in a line form an upper semilattice. Every two of these lines have the shortest line with both embedded in it as their least upper bound. But two of these lines have a greatest lower bound only if there is a common line embedded in them.

With lines and the embedding relation ready for use, shapes follow immediately. A shape is given by a finite but possibly empty set of lines. Not every set of lines will do, however, if different sets of lines are always to determine different shapes. Take two squares side-by-side. This shape may be determined in many ways, say, by seven lines: the shared side and the six other sides of the squares, or by five lines: the two horizontals and the three verticals. The lines in a shape must be chosen with care if a single shape is not to be taken for many. Maximal lines are used to ensure that sets identify shapes uniquely. Maximal lines are such that two can be embedded in exactly the same lines only if there is a line embedded in the shortest of these that has no line embedded in it that is embedded in either of the two lines. If maximal lines are collinear, then they are separated by a gap; otherwise, they may touch or intersect. Every shape is determined by a set of maximal lines. This set contains the smallest number of longest lines that combine to make the shape.

As a set, a shape has limited possibilities for decomposition into elements and subsets. But the maximal lines in the shape have other lines embedded in them that may be maximal in combination. In this case, they determine the shapes that are parts of the shape, and that may be used to decompose it. The subshape relation fixes this idea formally: one shape is a subshape of another shape if every maximal line in the first is embedded in some maximal line in the second. Two squares side-by-side are determined by a set of five maximal lines. This shape has a rectangle and two squares among its many subshapes. The rectangle is formed by combining some of the maximal lines in the shape and is given by a subset of these. Neither of the squares, however, is given by any such subset; both have maximal lines that are not in the shape but that are embedded in maximal lines that are. And other subshapes may be such that none has any maximal line that is a maximal line in the shape.

Just as the embedding relation does for lines, the subshape relation provides every shape with a definite structure given by all of its subshapes. These are partially ordered by the subshape relation and form a Boolean algebra. Similar shapes always have the same structure.
The subshape relation allows for shapes to be decomposed in any way whatsoever. Manipulating shapes goes forward reciprocally with two operations to combine shapes. Both sum and difference are readily defined in terms of the subshape relation, or in terms of reduction rules that apply to sets of lines to produce sets of maximal lines. The latter approach is constructive and is worth a closer look.

Every two shapes $A$ and $B$ have a unique sum $A + B$ that corresponds to drawing the shapes as one. The set of maximal lines for $A + B$, however, need not contain any of the maximal lines in $A$ or in $B$. For example, two squares side-by-side have two maximal lines apiece that are not maximal lines in their sum. But in pairs -- one from each square -- these lines form maximal lines that are. Reduction rules combine lines to produce maximal lines; they are applied recursively, as long as possible, beginning with the union of the set of maximal lines for $A$ and the set of maximal lines for $B$ to obtain the set of maximal lines for $A + B$. Reduction rules specify the details in this procedure: if the lines $l$ and $l'$ are such that a common line is embedded in both or both share an endpoint and are embedded in a common line, then replace $l$ and $l'$ with the shortest line that has both embedded in it.

It is interesting to notice that the reduction rules for sum allow for shapes to be defined in schemata. This establishes families of shapes that share special features, and the basis for parametric variation as a way to obtain different shapes. A shape schema $A(x)$ is a set of variables. Lines are assigned to these by a function $g$, and may be required to satisfy certain conditions. The set $g(A(x))$ produced in this way, however, may not determine a shape, as the lines in it may not be maximal. But once the reduction rules for sum are applied to the set, the shape $R(g(A(x)))$ is given.

Where sum adds shapes together, difference subtracts one shape from another. Every pair of shapes $A$ and $B$ in this order has a unique difference $A - B$ that corresponds to erasing those subshapes of $A$ that are also subshapes of $B$. If either square is subtracted from two squares side-by-side, then another shape made up of three sides of the other square is produced. The set of maximal lines for $A - B$ is obtained from the set of maximal lines for $A$. Beginning with this set, reduction rules are applied recursively, once again as long as possible. The rules specify the details in this procedure: if a line $l$ is embedded in a maximal line $l'$ in $B$, then remove $l$; but if this is not so and $l$ and $l'$ have a common line embedded in them, then replace $l$ with one or two lines, so that each has an endpoint of $l$ and is the longest such line embedded in $l$ that has no line embedded in it that is embedded in $l'$. 


The algebra of shapes \( U^* \) with the operations of sum and difference is equivalent to a Boolean ring with a zero but no unit. The transformations, too, are used in \( U^* \) to change shapes into similar shapes, maximal line-by-maximal line, and they distribute over both sum and difference. In this case, with the trivial exception of the set containing the zero only, \( U^* \) has no proper subset with exactly the same properties.

Computations in \( U^* \) are many and varied. A computation is a series of shapes produced by following rules that change shapes into shapes. These changes are carried out recursively by drawing and erasing lines. A rule \( A \rightarrow B \) defines a sequence of these operations with the transformations and with sum and difference, so that an occurrence of one shape \( A \) is replaced with the corresponding occurrence of another shape \( B \). The rule applies to a shape \( C \) whenever

\[
(1): \text{ there is a transformation } t \text{ that makes } A \text{ a subshape of } C
\]

to produce a new shape according to the formula

\[
(2): \ (C - t(A)) + t(B).
\]

To replace shapes in this way exploits the full resources of \( U^* \) and thereby provides the basis for every computation with shapes. These computations are of all sorts. In fact, every computation defined in a Turing machine can be carried out equivalently by following rules in \( U^* \).

Like shapes, rules may be defined in schemata. This is an important generalization that combines replacement and parametric variation, and so greatly facilitates writing rules that are of broad use. A rule schema \( A(x) \rightarrow B(x) \) is a pair of shape schemata \( A(x) \) and \( B(x) \) that may share variables. If \( g \) is a function that assigns lines to all of the variables in both of these schemata, and the reduction rules for sum are used to ensure that all lines are maximal, then the rule \( R(g(A(x))) \rightarrow R(g(B(x))) \) is defined.

Some small effort was required to characterize the algebra of shapes \( U^* \), and to provide the means to define computations in it. This effort, however, is amply repaid. Drawing with lines is now as rigorous as doing arithmetic, and is no more mysterious. And drawings themselves may be viewed as active agents in computation, as well as descriptions in the usual way. But drawings alone rarely make a design; they are typically combined with symbols that work in conjunction with them. These important interactions require a new algebra that can be formed in the Cartesian product of \( U^* \) and the algebra of sets of labeled points \( V^* \).
A labeled point is defined when a label from a specified set, say, the alphabet \( a, b, c \), ..., is associated with a point. Labels may be just that, in which case, they are used to identify and classify particular points, or they may have a semantics allowing them to carry important information about shapes and about other things. A labeled point may be transformed to produce a new one in which the same label is associated with another point. Finite but possibly empty sets of labeled points are partially ordered by the subset relation, and are combined by union and relative complement to form the algebra \( V^* \). Like \( U^* \), \( V^* \) is equivalent to a Boolean ring with a zero but no unit. And if the transformations are used in \( V^* \), then they distribute over both of its operations.

It is convenient to define sets of labeled points in schemata. Each schema \( P(y) \) is a set of variables. Labeled points are assigned to these by a function \( g \), and may be required to satisfy certain conditions. Further, pairs of schemata \( P(y) \land Q(y) \) possibly with common variables define rules to carry out computations in \( V^* \). Each rule determines a sequence of operations allowing for replacement, and applies recursively as specified in (1) and (2), where \( A, B, \) and \( C \) now stand for sets of labeled points, and subshape, sum, and difference correspond to subset, union, and relative complement, respectively. And notice that rules in \( V^* \), too, are enough to carry out any computation defined in a Turing machine.

Lines and labeled points are combined to make labeled shapes. A labeled shape \( \langle A, P \rangle \) has two components -- a shape \( A \) and a set of labeled points \( P \) -- and is an element in the algebra \( U^*V^* \) formed in the Cartesian product of the algebras \( U^* \) and \( V^* \). The relation and operations on \( U^*V^* \) combine subshape with subset, sum with union, and difference with relative complement. If these combinations are named by the shape relation or operation involved, then the labeled shape \( \langle A, P \rangle \) is a subshape of the labeled shape \( \langle B, Q \rangle \) whenever the shape \( A \) is a subshape of the shape \( B \), and the set of labeled points \( P \) is a subset of the set of labeled points \( Q \). And, for example, the sum of these labeled shapes has as components the sum of \( A \) and \( B \), and the union of \( P \) and \( Q \). A transformation \( t \) of the labeled shape \( \langle A, P \rangle \) is the labeled shape \( \langle t(A), t(P) \rangle \). The algebra \( U^*V^* \) shares the properties of the algebras \( U^* \) and \( V^* \); it is equivalent to a Boolean ring, and the transformations distribute over its operations.

Computations in \( U^*V^* \) are defined in shape grammars. Shape grammars have been used in architecture for many years to define relations containing designs. These relations are called languages. Shape grammars have been used to define languages for Palladio, Wren, Wright, and Terragni, and for vernacular styles from different
periods and places. Shape grammars provide designs for churches, villas, houses, multi-storied structures, and buildings of other types, for ornament to decorate them, for furniture to fill them, and for gardens in which to put them. And new shape grammars are always forthcoming; they have been worked out from scratch to explore fresh ideas, and by combining or transforming shape grammars already in use to profit from past experience with established architectural styles and known building types.

A computation in a shape grammar is a series of labeled shapes produced by following rules that change labeled shapes into labeled shapes. Each rule combines a rule in $U^*$ and a rule in $V^*$ that are meant to apply together in $U^*V^*$, the one to the shape and the other to the set of labeled points in a labeled shape. In this way, the computation in the shape grammar combines two computations -- one with shapes and one with sets of labeled points -- that are carried out in parallel and influence one another mutually. These interactions vary according to how rules in $U^*$ and rules in $V^*$ are linked in $U^*V^*$. Rules in $U^*V^*$ establish dependencies among shapes and sets of labeled points that are propagated within labeled shapes and from labeled shape to labeled shape as the rules are defined in schemata and then applied. The mechanism that makes this possible is already familiar.

In a shape grammar, a rule schema $<A(x), P(y)> \rightarrow <B(x), Q(y)>$ combines schemata for shapes $A(x)$ and $B(x)$, and schemata for sets of labeled points $P(y)$ and $Q(y)$. Variables may be shared among these schemata, and conditions on values assigned to variables may relate lines and labeled points. A rule is defined whenever a function $g$ is used to assign appropriate values to all of the variables in the schema $<A(x), P(y)> \rightarrow <B(x), Q(y)>$, and the reduction rules for sum are used to determine shapes from the sets $g(A(x))$ and $g(B(x))$. The rule is applied as specified in (1) and (2), this time interpreted for labeled shapes.

The idea of combining algebras in Caretsian products, so that computations can be carried out in parallel, has applications that go beyond shape grammars as they are typically used to define languages of designs. For example, $U^*$ and $V^*$ can be combined in repeated products to define relations containing designs that have as components plans, sections, and elevations, or descriptions for the separate parts of an assembly. Consider a relation for building plans and elevations in a certain style. The relation can be defined in the product $U^*V^*U^*V^*$, where plans are labeled shapes produced in the first instance of $U^*V^*$ and elevations are labeled shapes produced in the second. In this case, rules in two shape grammars are combined, allowing for two computations with labeled shapes to be performed in parallel. Or
the relation can be defined in an analogous fashion in the product $U^*U^*V^*$ whenever
the labels used in plans and the labels used in elevations are taken from disjoint sets. It
is easy to imagine how schemata might be used in both of these new algebras to
combine rules for plans and rules for elevations.

Suppose an opening is to be made in a wall of a plan. If the wall is an exterior one,
then a corresponding opening is required in the elevation; otherwise, the elevation is
unchanged. In the first case, a rule for plans must be combined with a rule for
elevations. If the location and the width of the opening in the plan and the location and
the width of the opening in the elevation are to correspond, then, in the schema in
which rules for plans and elevations are combined, some of the variables given to
define rules for plans and some of the variables given to define rules for elevations
must be shared, or conditions must be given to establish dependencies among them.
The height of the opening, however, need not be related to its width and may vary
freely, allowing for some of the variables to be independent of others. In the second
case, a rule for plans and an identity defined with the zero in $U^*V^*$ or in $U^*$ are
combined. The plan can so be changed without changing the elevation.

Not everything in a design, however, is always a shape from $U^*$ or a set of labeled
points from $V^*$. Other devices are usually required, and are readily available in other
algebras. For example, a labeled shape in a design may be described in the design to
clarify intention or to facilitate some kind of analysis. Typically, this involves one or
more decompositions. Each describes the labeled shape in terms of a list (alphabet or
vocabulary) of labeled shapes, and is given by a finite, nonempty set of
transformations of labeled shapes in the list that sum to make the labeled shape. This
decomposition and others belong to an algebra that is a Boolean ring like $V^*$, only this
time defined on sets of labeled shapes instead of on sets of labeled points. Every set in
the algebra except the zero is a decomposition of some labeled shape, and every
labeled shape in $U^*V^*$ has at least one decomposition and even infinitely many of
them in the algebra. Computations with decompositions are defined in set grammars
in which spatial relations are used to express compositional ideas. The rules in
a set grammar are pairs of nonempty sets of labeled shapes that may be defined in
schemata; they are applied as specified in (1) and (2). And when the algebra for sets
of labeled shapes is combined in Cartesian products with $U^*$ and $V^*$, shape grammars
and set grammars can be used together to define relations among labeled shapes and
their decompositions.

There are more ways to ramify designs, too. The labeled shapes in decompositions
may be ordered hierarchically or in some other way, and assigned to special
categories. In addition to labeled shapes and their decompositions, designs may
contain other geometrical representations based on planes or on solids. And designs may contain graphs and strings of symbols. Opportunities for algebras in which relations containing designs can be defined in computations are vast, and can always be multiplied in yet another Cartesian product.

The use of algebras and computations to define relations containing designs shows the kind of virtuosity that is to be encouraged in intelligent practice, and it can show something about practice as well. Attempts to define and use relations containing designs suggest that familiar slogans like "form follows function" that are intended to guide practice are without foundation, that simple appeals to programme, context, technology, or material as the exclusive generator of form are without force, and that formal, functional, rational, and historical accounts of how designs come to be are partial and incomplete. Attempts to define and use relations containing designs in which drawings, descriptions, and devices of other kinds are linked and work together suggest that actual practice is more complex. Designs follow from a confluence of considerations, some dealing with form, and others dealing with function, material, construction, and so on. The impetus for practice comes from multiple perspectives that may each be dominant at different times. These perspectives are connected, allowing for interests and goals in different domains to interact, each able to influence the others and for a time to direct the way to go. Designs are complicated and multifaceted; they are the very stuff of relations, algebras, and computations.

Background. Many of the ideas presented in this paper were originally considered in Stiny (1975). A detailed, up-to-date account will appear in Stiny (in preparation). For now, an introductory review of work on shape grammars is given in Stiny (1985), a brief account of spatial relations and set grammars in Stiny (1982), and a worked example with a relation containing designs in Stiny (1981). An algebra for geometrical representations based on planes and its use with $U^*V^*$ to define computations with shapes and colors are considered in Knight (in press), and a good discussion of geometrical representations based on solids is presented in Requicha (1980).
REFERENCES

Knight, T. W., in press, "Color Grammars: Designing with Lines and Colors", Planning and Design: Environment and Planning B.


Stiny, G., in preparation, Shape: A Primer in Algebra, Grammar, and Description.