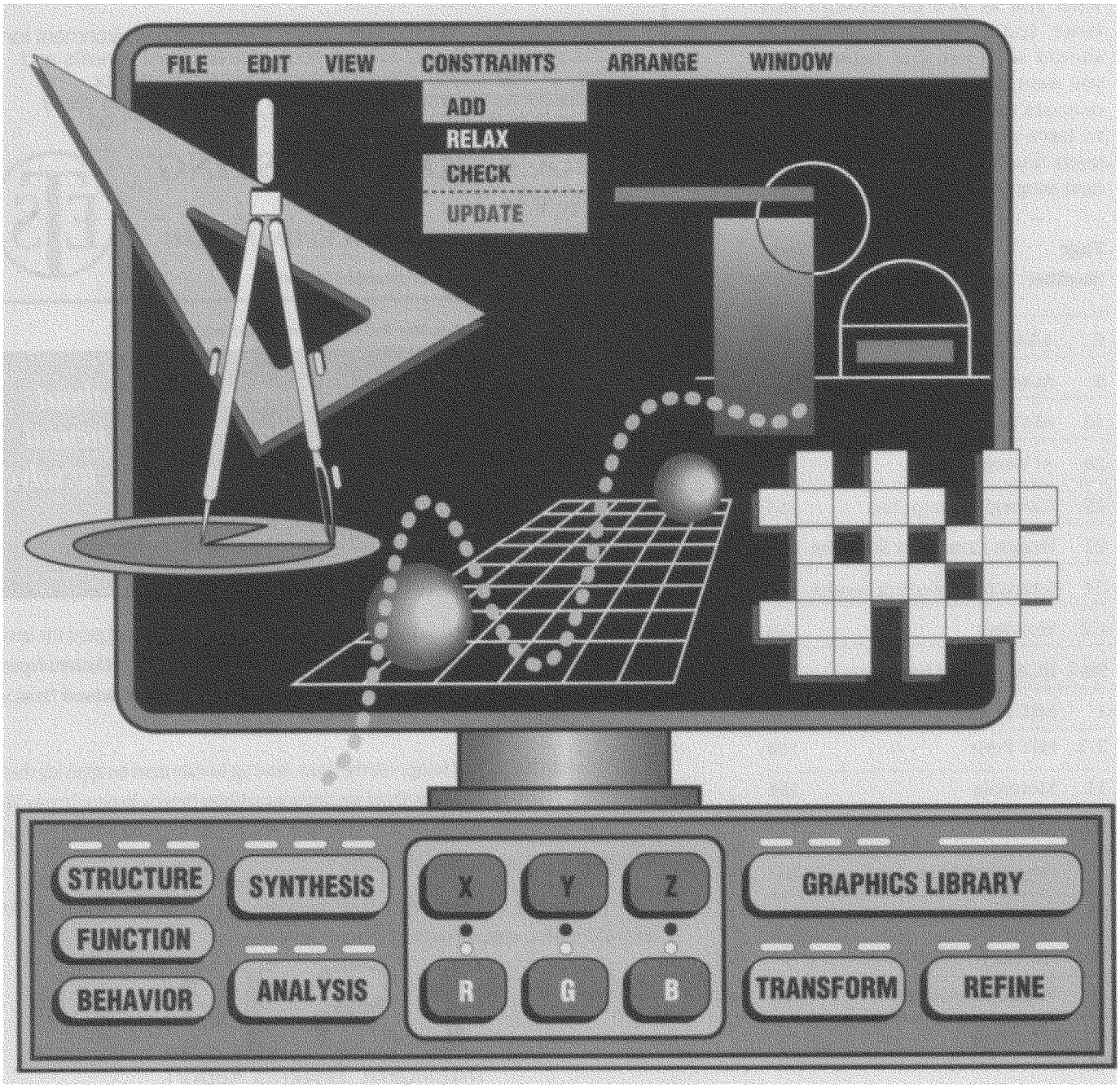
AI Magazine Volume 11 Number 4 (1990) (© AAAI)

Design Prototypes: A Knowledge Representation Schema for Design

John S. Gero



This article begins with an elaboration of models of design as a process. It then introduces and describes a knowledge representation schema for design called design prototypes. This schema supports the initiation and continuation of the act of designing. Design prototypes are shown to provide a suitable framework to distinguish routine, innovative, and creative design.

Designers are change agents in society.

Although there are designers who claim design is a mysterious activity not amenable to scientific examination, research into design continues Although there are publications by designers on how to design dating back to Roman times, notably by Vitruvius, the nineteenth-century design thinkers actually began work on articulating design as a process (Durand 1802). However, it was not until the 1960s that major research programs were initiated. These programs were originally founded on the systems view and used concepts from operations research (Jones and Thornley 1963). More recently, information-processing models founded on AI concepts have provided an impetus for renewed research into design in its various aspects (Simon 1969; Coyne et al. 1990). Many foundational ideas in AI are proving to be useful in developing formal models of design as an activity.

What Is Design?

Designers are change agents in society. Their goal is to improve the human condition in all its aspects through physical change. Although design for many continues to remain a mysterious activity, it has been recognized as an important activity for more than 4000 years In approximately 2000 BC, Hammurabi, king of Babylon, enacted a law that both recognized design and made it dangerous (figure 1). Design research has a number of goals, including a better understanding of design, the development of tools to aid human designers, and the potential automation of some design tasks.

Design appears to be carried out differently from the way we are taught to understand the world, which is largely derived from the Greek view of the world. Science has been developed as a means of attempting to explain and understand the world around us. It begins with a description of the world (which in itself is not a trivial act to produce) and some behaviors and attempts to produce causal dependencies between them. Science then can be used to attempt to produce a purpose for the world. Design exists because the world around us does not suit us, and the goal of designers is to change the world through the creation of artifacts. Designers design by positing functions to be achieved and producing descriptions of artifacts capable of gener-

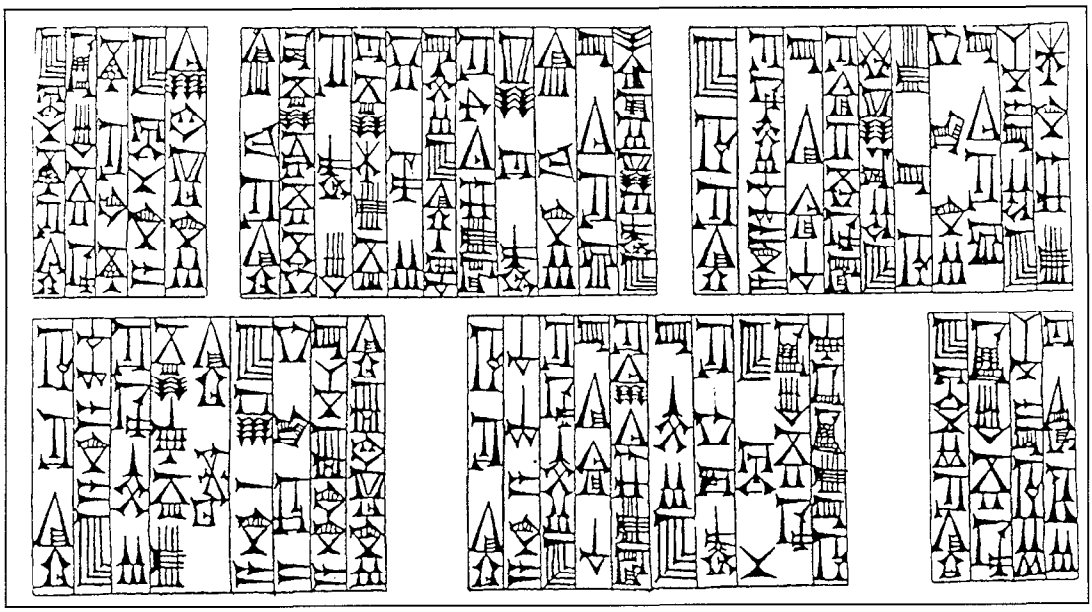


Figure 1. Hammurabi's Code, from an Engraving on a Stella in Cuneiform in the Louvre, Paris.
"If a designer-builder has designed-built a house for a man and his work is not good, and if the house he has designed-built falls in and kills the householder, that designer-builder shall be slain "

0738-4602/90/$4 00 ©1990 AAAI

Design activity can be now characterized as a goal-oriented constrained, decision-making exploration, and learning activity . . .

ating these functions. In this sense, design is the opposite of the traditional scientific explanation.

Thus, design is purposeful, and the activity of designing is goal oriented. The metagoal of design is to transform requirements, generally termed functions, which embody the expectations of the purposes of the resulting artifact, into design descriptions.

The result of the activity of designing is a design description. This design description is generally represented graphically, numerically, or textually. The purpose of such a description is to transfer sufficient information about the designed artifact so that it can be manufactured, fabricated, or constructed.

A prevalent and pervasive view of designing is that it can be modeled using variables and decisions made about what values should be taken by these variables. The activity of designing is carried out with the expectation that the designed artifact will operate in the natural world and the social world. These worlds impose constraints on the variables and their values; so, design could be described as a goal-oriented, constrained, decision-making activity. However, design distinguishes itself from other similarly described activities not only by its domain but also by additional necessary features. Designing involves exploration, exploring what variables might be appropriate. The process of exploration involves both goal variables and decision variables. In addition, designing involves learning: Part of the exploration activity is learning about emerging features as a design proceeds. Finally, design activity occurs within two contexts: the context within which the designer operates and the context produced by the developing design itself. The designer's perception of what the context is affects the implication of the context on the design. The context shifts as the designer's perceptions change. Design activity can be now characterized as a goal-oriented, constrained, decision-making, exploration, and learning activity that operates within a context that depends on the designer's perception of the context.

Models of Design

The purpose of designing is to transform function F (where F is a set) into a design description D in such a way that the artifact being described is capable of producing these functions. For example, when designing windows, some of the functions include the provision of daylight, control of ventilation, and access to a view. The design description would take the form of drawings and notes. Thus, a naive model of design is

$F \rightarrow$ D,

where $\rightarrow$ is some transformation. There is, however, no direct transformation capable of achieving this result.

A design description represents the artifact's elements and their relationships; it is labeled structure S. In the window design example, the artifact's elements are the glazing and the frame and their topology. Computer-aided drafting systems have become the means by which structure is transformed into a design description; that is,

S $\rightarrow$ D.

Another model of design is

$F \rightarrow$ S.

Here, a transformation does occasionally exist in the form of a direct mapping between function and structure, often termed *catalog lookup*. This transformation occurs at the element level of an artifact and is not considered designing. More generally, no direct transformation between function and structure exists, which leaves a requirement for an indirect transformation between function and structure.

Function has been defined in another context as the relation between the goal of a human user and the behavior of a system (Bobrow 1984). In designing, behavior can be viewed in two ways. First is the behavior of the structure B_S (where B_S is a set), which is directly derivable from structure:

S $\rightarrow B_S$.

In the window design example, the behaviors of the structure include the light flux transmitted, the ventilation rate, and the various solar gains. This process is that of *analysis* and presupposes the delineation of which behaviors to determine.

Transforming function to expected behaviors B_e (where B_e is a set) provides the second view of behavior. The expected behaviors for the window design example include light transmission, ventilation rates, and solar collection. The expected behavior provides the syntax by which the semantics represented by function can be achieved:

$F \rightarrow B_e$.

This process is that of *formulation* or *specification* in design. The predicted behavior of the structure can be compared with the expected behavior required to determine if the structure synthesized is capable of producing the functions:

$B_e \leftrightarrow B_S$,

where $\leftrightarrow$ is a comparison. This comparison process is termed *evaluation* in design.

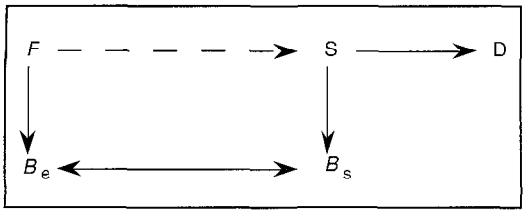


Figure 2. Model of Design as a Process.

B_e = Set of expected behaviors
B_s = Set of actual behaviors D = Design description
F = Set of functions S = Structure
→ = Transformation
---> = Occasional transformation
<-> = Comparison

Another model of design is
$F \rightarrow B_e$
$B_e \rightarrow S(B_s)$.
Here, the function is transformed to expected behavior. This expected behavior is used in the selection and combination of structure based on a knowledge of the behaviors produced by this structure. This process is called *synthesis.*

When structures are synthesized, they produce their own behaviors, which can be a useful superset of the expected behaviors. This process can change the range of expected behaviors and through them the function being designed for, leading to a *reformulation.* Reformulation can also occur when the evaluation of the comparison between the behavior of the structure and the expected behavior is unsatisfactory and cannot be made satisfactory by manipulating the structure. This reformulation leads to a change in expected behavior.

Figure 2 shows how these transformations appear in design. A general model of design as a process involves the following activities (figure 3): formulation, synthesis, analysis, evaluation, reformulation, and production of design description.

Two fundamental research issues present themselves here: What appropriate representation frameworks are for design knowledge and what appropriate transformation processes are for design. The remainder of this article only addresses the first of these.

The Conceptual Schema "Design Prototypes"

Schemas are introduced here, and the conceptual schema "design prototypes," which is used to represent design knowledge, is elaborated and described.

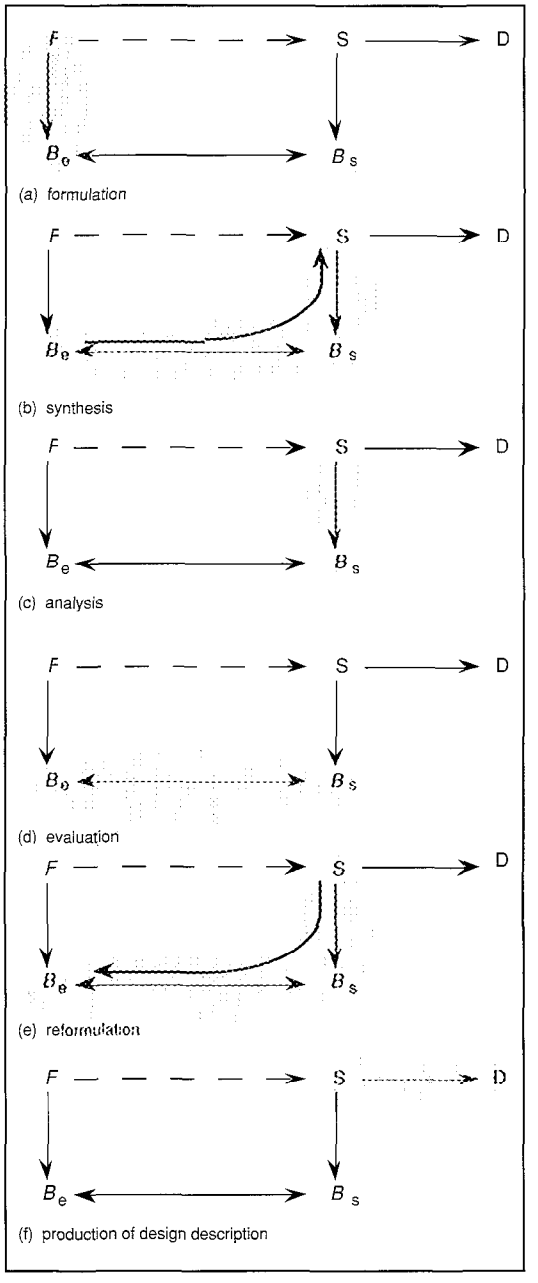


Figure 3. Activities in Design.

Schemas

How is it that designers can begin designing with incomplete information and before all the relevant information is available? Indeed, because design is an exploration process, what is relevant only manifests itself as the design proceeds and varies with the decisions taken. Where do structures come from? How do additional functions appear as a design progresses? How does a designer know what behaviors to analyze for? It is suggested that

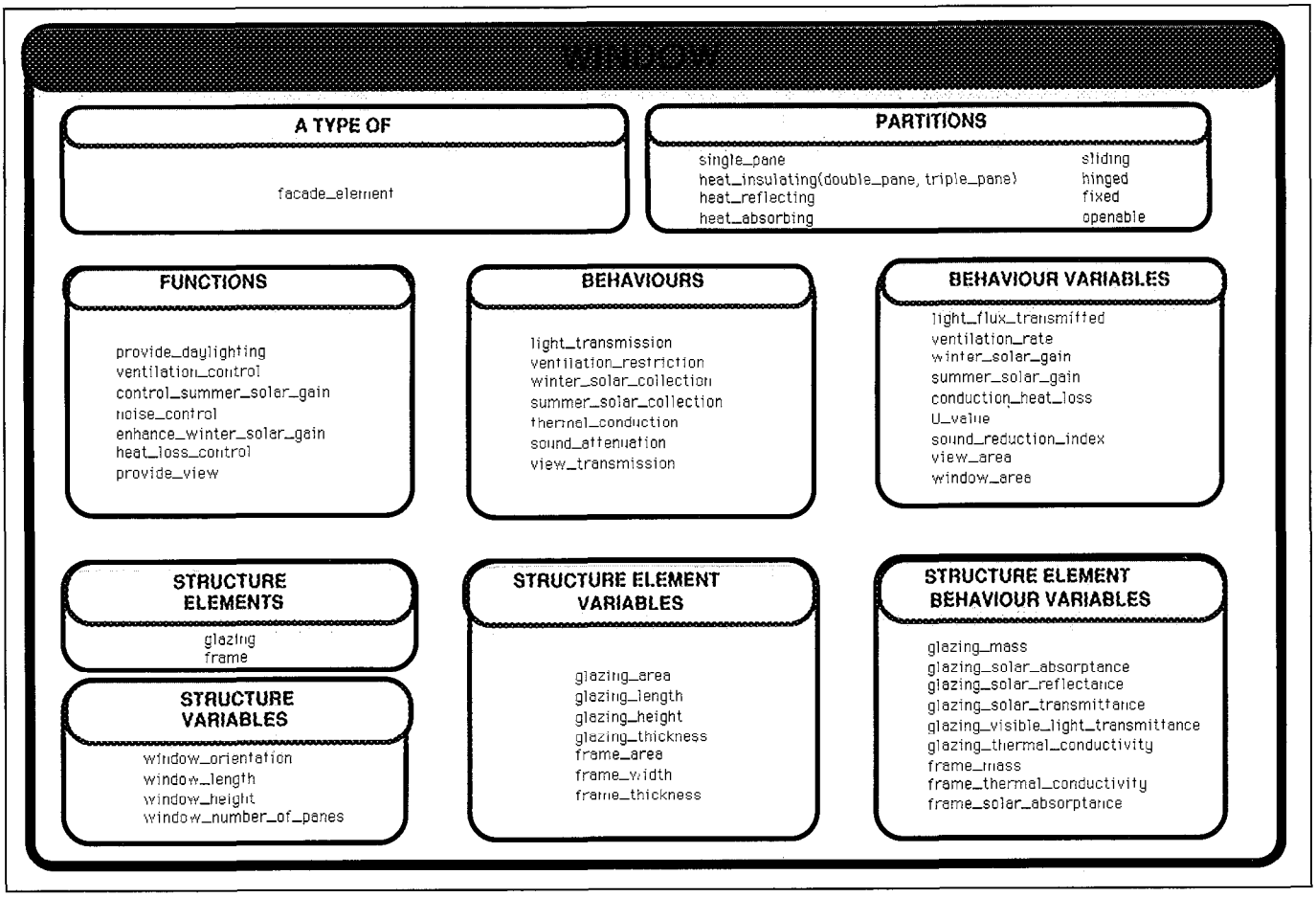


Figure 4a. Diagrammatic representation of the function, behavior and structure components of the design prototype schema for window design (Tham et al., 1990).

human designers form their individual design experiences into generalized concepts or groups of concepts at many different levels of abstraction; that is, they schematize their knowledge. Such schemas consist of knowledge generalized from a set of alike design cases and form a class from which individuals can be inferred. In design, any schema must at least be able to incorporate function, structure, behavior, and design description and be accessed by elements within these components. It will be seen later that more than these elements needs to be incorporated within a schema. No satisfactory schemas appear to be available for the representation of this generalized heterogeneous design knowledge that have sufficient expressiveness and power. None can be used to explain how a design begins and continues or how design processes can be categorized.

It is customary to talk about types as a means of classifying the world. There are important distinctions between archetypes, stereotypes, and prototypes. *Archetypes* are the first and often singular examples of their type. Thus, the Taj Mahal in India is the archetypal romantic architectural setting, and a red Ferrari is the archetypal rich, young bachelor's car. In both cases, there can be no substitute. Each archetype might provide an analog for another design, but reproducing the Taj Mahal would not be considered a feat of designing. *Stereotypes* are copies without change. Mass production of goods is a means of stereotyping. In designing, stereotyping (that is, the process of reproduction) belies the design endeavor. *Prototypes* are the first on which others are modeled. This is one of the bases of the notion of design prototypes being elaborated here within the context of knowledge-based design.

Design Prototypes

A *design prototype* (Gero 1987) is a conceptual schema for representing a class of a generalized heterogeneous grouping of elements derived from alike design cases that provides

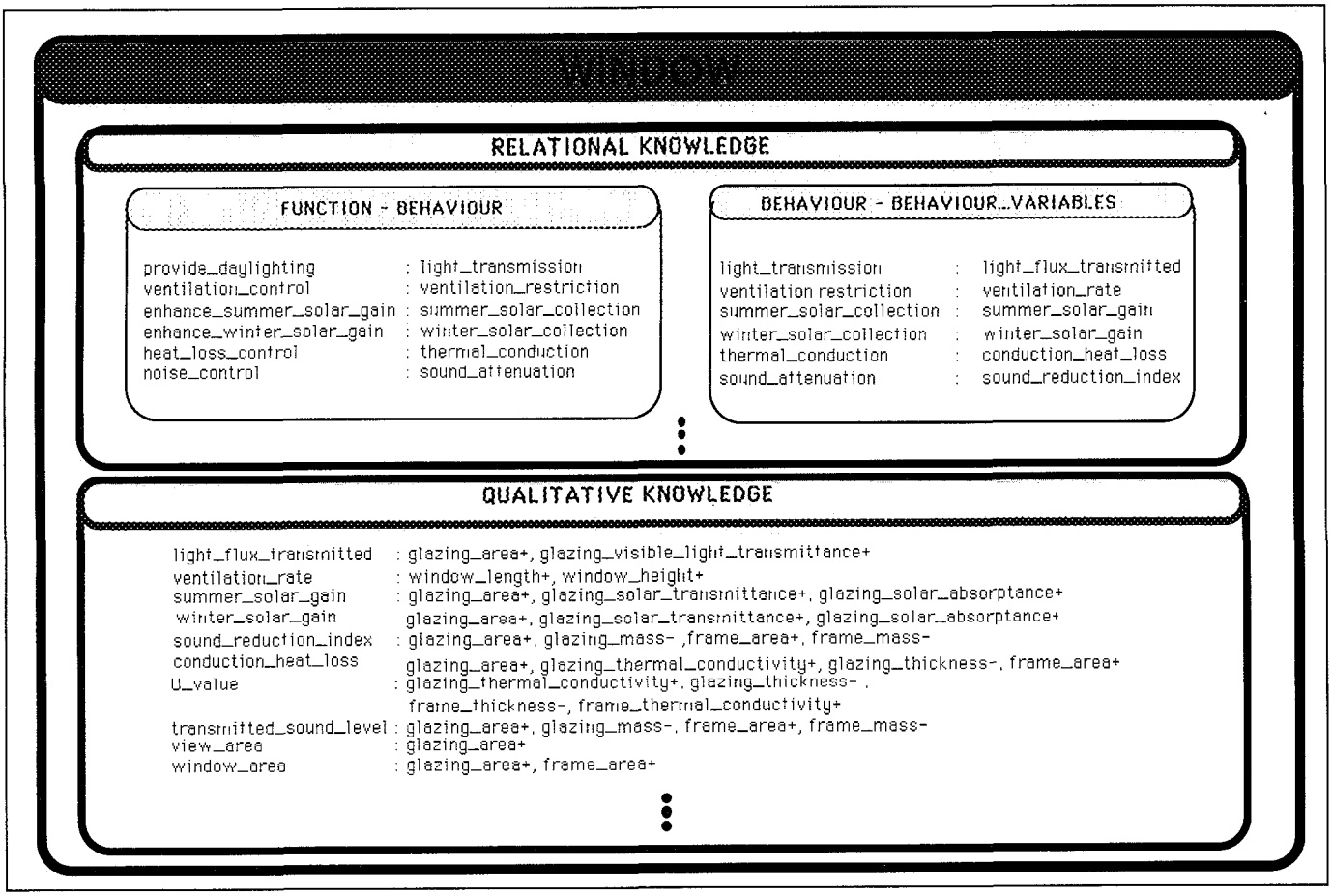


Figure 4b. Diagrammatic representation of the relational and qualitative knowledge of the design prototype schema for window design (Tham et al., 1990).

the basis for the start and continuation of a design. Design prototypes provide this basis by bringing all the requisite knowledge appropriate to the design situation together in one schema.

A designed artifact can be broadly interpreted in terms of the three variable groups of function, structure, and behavior. The level of specificity in each of these groups depends on the granularity and level of abstraction being represented. Thus, at an early stage of designing, an appropriate design prototype might primarily contain function and behavior, with little information on structure, but at a later time, an appropriate design prototype might contain considerable detail in the structure group. A design prototype brings together these three groups and the relations between them, which includes processes for selecting and obtaining values for variables Design prototypes draw from such sources as prototype theory (Osherson and Smith 1981) and scripts (Schank and Abelson 1975). *Prototype theory* construes membership of a concept to be determined by its similarity to the concept's best exemplar. Design prototypes use the notions of generalization to produce the prototype. Although closely related to scripts, design prototypes include semantics and are not bound by time sequence.

Although it is well recognized that there is no function in structure and that there is no structure in function, human and design experience produces a connection between function and structure. Once this connection is learned, it is difficult to unlearn. Once the connection between behavior and structure is made, and the connection between behavior and function is made, these connections form the basis of much of a designer's knowledge. Function, structure, behavior, and relationships form the foundation of the knowledge that must be represented for specific design processes to be able to operate on them. In natural discourse, the distinction between function and structure sometimes becomes blurred to the extent that the label of the structure takes on the meaning of the func-

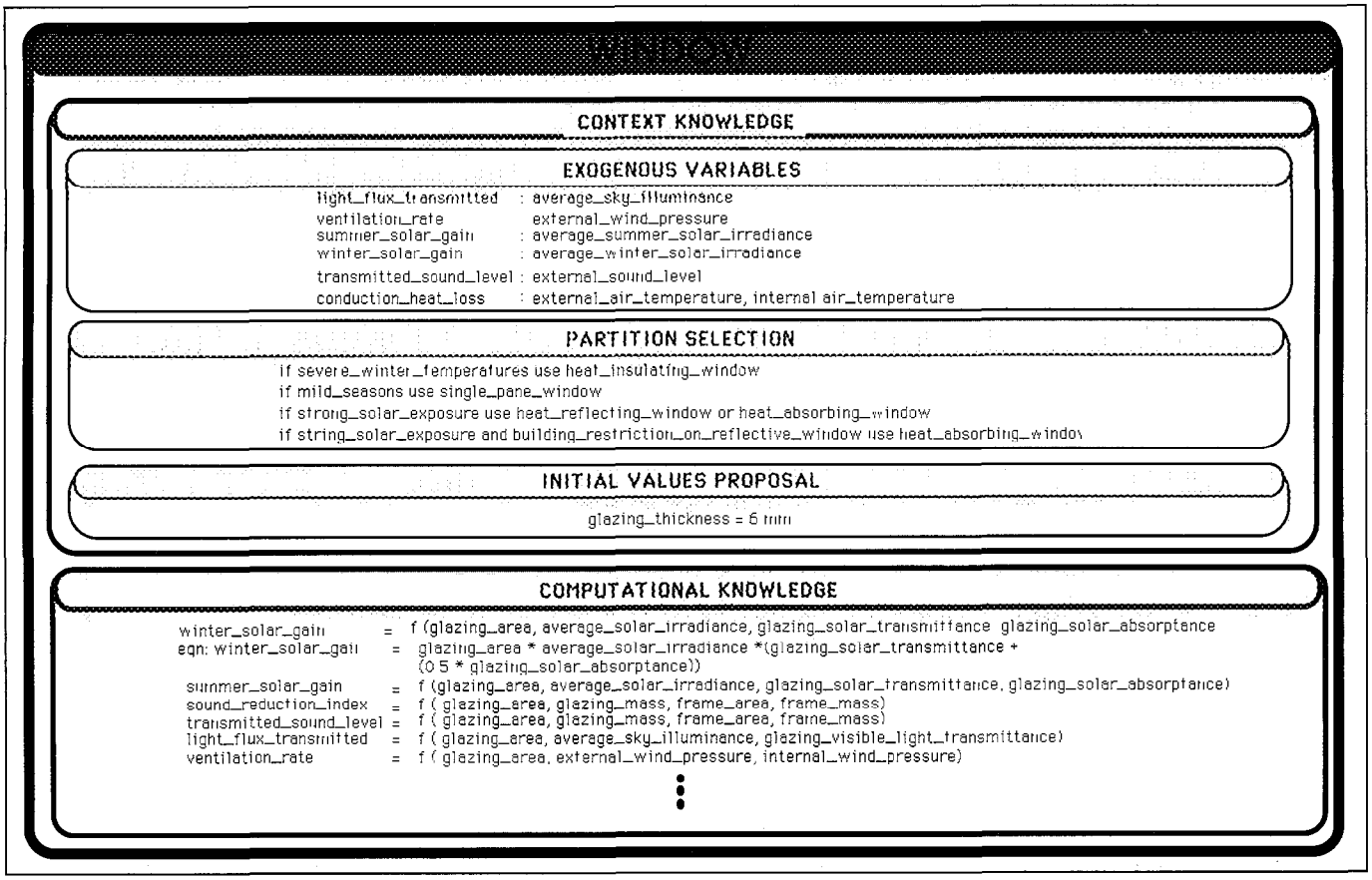


Figure 4c. Diagrammatic representation of the context and computational knowledge of the design prototype schema for window design (Tham et al., 1990).

tion. For example, the label of a particular copier, Xerox™, is slowly taking on the meaning of its function, that is, to copy. However, if reasoning is to occur in transforming function to structure, then a clear separation must be made between them and between function, structure, and behavior.

Structure of Design Prototypes

A design prototype separates function (F), structure (S), expected behavior (B_e) and actual behavior (B_s). It also stores relational knowledge between them (K_r) as well as qualitative knowledge (K_q), computational knowledge (K_c), and context knowledge (K_{ct}).

Relational knowledge provides and makes explicit the dependencies between the variables in the function, structure, and behavior categories and can take the form of a dependency network. Relational knowledge identifies the relevant variables in going from function to behavior and behavior to function and from behavior to structure and structure to behavior. Relational knowledge allows for the specialization of the information in a prototype to a specific design situation (see Designing Using Design Prototypes).

Qualitative knowledge (a subset of qualitative reasoning) is an adjunct to relational knowledge and provides information on the effects of modifying values of structure variables on behavior and function. Included here are the normal ranges of values of variables found in the generalization. Qualitative knowledge can be used to guide any decision-making process.

Computational knowledge is the quantitative counterpart of qualitative knowledge and specifies symbolic or mathematical relationships among the variables. Computational knowledge is used to determine values of variables.

Constraints appear in both qualitative knowledge and computational knowledge. Constraints on function appear as expected behaviors; constraints on structure reduce the range of possibilities.

Context knowledge identifies the exogenous variables for a design situation and specifies that values for these variables must come from outside the design prototype, that is, from the context (C).

In addition, there is knowledge concerning the design prototype itself (K_p). This knowledge comprises the typology (T) of the design prototype, which identifies the broad class of which the design prototype is a member, and partitions (P) represent the subdivisions of the concept represented by the prototype. Partitioning a design prototype supports viewing it from many perspectives. Once the partition or combination of partitions is selected, only information pertaining to these partitions is made available. In this sense, partitioning of design prototypes ultimately reduces the space of potential designs.

A design prototype, P, can be symbolically represented as

P = (F, B, S, D, K, C) ,

where

B = (B_s, B_e)

C = context

K = (K_r, K_q, K_c, K_{ct}, K_p)

K_p = (T, P)

P = partition

T = typology.

Figure 4 is a diagram of the design prototype schema for the knowledge associated with window design.

In summary, a design prototype brings together all the requisite knowledge appropriate to a specific design situation. Although the contents of a design prototype are developed by individual designers, like-minded designers will tend to agree on its general contents. Thus, a design prototype concerned with the initial design of a house is likely to include such notions as style, location on site, orientation, existence of spaces based on their functional activities, and building proposal codes. A designer will draw on many design prototypes during the course of developing any design.

Designing Using Design Prototypes

Designing using design prototypes can be thought of as matching a cognitive view of a process model of design. Studies of designers indicate that they link function and structure and select concepts to follow early in a design (Lawson 1980). Later in this article, I present the specifics of routine and nonroutine design. Here, I give a descriptive outline of designing using design prototypes.

A designer begins with required functions from a client. Sometimes clients also specify required structures. These requirements are used to retrieve potentially useful design prototypes on the basis that they are indexed by these requirements. These retrieved design prototypes represent the set of concepts that a designer remembers when s/he examines the requirements. Each design prototype contains more function and structure (and behavior) than were used to index it. This is similar to being reminded of additional related functions and structures. In this way, design prototypes provide a means by which given a little situational information, potentially appropriate concepts are retrieved, and the designer has available a fleshed-out set of concepts that can lead in many directions. However, not all retrieved design prototypes are likely to be equally useful, and they need to be evaluated and one or more selected. Once a design prototype is selected, an instance is created. Each *instance* represents the beginning of a design alternative. Instances are subsets of their design prototypes. They initially inherit the entire structure of the design prototype, but not all the knowledge in the design prototype might be useful in the particular context; so, it is pruned and this pruning propagated using the dependency knowledge.

The pruned instance is now the equivalent of a formulated design problem at this level of abstraction and granularity. It contains default values and normal ranges of values for variables. The various types of knowledge are used to attempt to determine specific values for variables.

If there is insufficient knowledge to produce values for specific variables, these variables are transformed into requirements. These requirements are then used to retrieve additional design prototypes that have the potential through their instances to produce these values, and the entire process is repeated. The use of default values provides a convenient means of controlling the propagation of design prototype retrieval down the scale of granularity. Thus, the designer does not need to know the detail when working at the overall level.

Designing using design prototypes allows for the start of a design at any level of available information. Because design prototypes carry a wide range of functions with them, retrieved design prototypes are a source of new functions. Retrieval by function alone still introduces structure, and retrieval by structure alone introduces functions associated with this structure. Design prototypes encode what the appropriate behaviors are to analyze for. Design prototypes provide a knowledge representation schema separate from the specific computational processes. (Computational processes that support design are described elsewhere [Coyne et al. 1990]). Design prototypes readily provide a framework that

. . . a design prototype brings together all the requisite knowledge appropriate to a specific design situation.

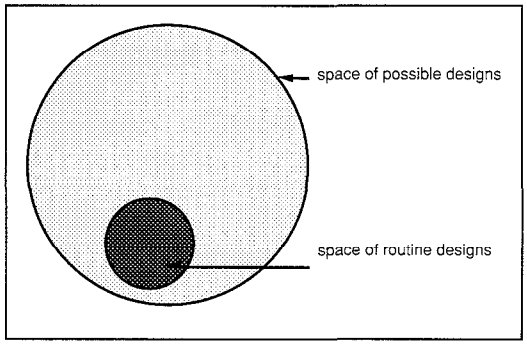


Figure 5. State Space of Routine Designs.

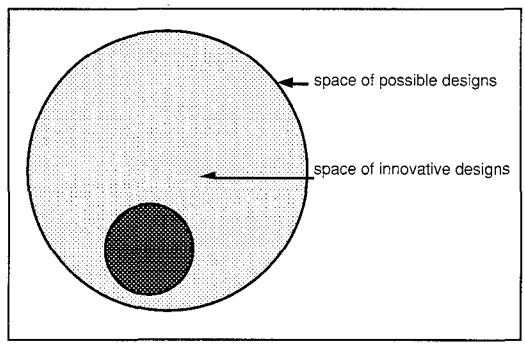


Figure 6. State Space of Innovative Designs.

supports both routine and nonroutine design, which are discussed in the next section.

Routine, Innovative, and Creative Design

There seems to be a general acceptance of the classification of design into routine, innovative, and creative (Brown and Chandrasekaran 1985; Coyne et al. 1987). Although there is argument about the definitions, they have proved to be useful.

Routine design can be defined as the design that proceeds within a well-defined state space of potential designs. That is, all the variables and their applicable ranges, as well as the knowledge to compute their values, are all directly instantiable from existing design prototypes (figure 5). In routine design, the space of designs produced is substantially smaller than the space of possible designs because of the constraints on the applicable ranges of values for variables.

Innovative design can be defined as nonroutine design that proceeds within a well-defined state space of potential designs. What distinguishes it from routine design is that the designs produced are outside the routine or normal space. This distinction is produced by manipulating the applicable ranges of values for variables. What results is a design with a familiar structure but novel appearance because the values of the defining variables are unfamiliar (figure 6).

Creative design can be defined as nonroutine design that uses new variables producing new types and, as a result, extending or moving the state space of potential designs. In the extreme case, a new and disjoint state space is produced. Creative design has the capacity to produce a paradigm shift (figure 7).

Routine Design

Routine design can be viewed as design prototype–instance refinement. Design prototypes are retrieved and selected, and instances are produced. These instances are refined in two ways. The first way is by pruning the set of variables to the applicable set through a specification of applicable functions, structures, or behaviors and propagating this specification. The second way is by determining the values of the applicable set of variables using the available knowledge. Figure 8 shows the outline of routine design by design prototype–instance refinement.

The specific processes used in qualitative knowledge and computational knowledge are separated from the representation of the design prototypes and from any system architecture. As stated previously, the processes cover formulation, synthesis, analysis, evaluation, reformulation, and production of design description. The use of a conceptual schema, such as design prototypes, which collects all the requisite knowledge, provides a basis for routine design.

Innovative Design

Innovative design can be viewed as design prototype–instance refinement with an adaptation of some of the knowledge concerning applicable ranges of variable values, that is, design prototype–instance adaptation. Additional processes for adaptation and the use of dependency knowledge to assist in the confirmation of the utility of any change are required.

Creative Design

Creative design, which involves the introduction of new variables into a design prototype, can be viewed as a means by which design prototypes are adapted to produce new design prototypes, that is, design prototype generation. In most cases, new prototypes are produced from old by changing them. It is possible to sufficiently adapt a design prototype so that the new design prototype is dis-

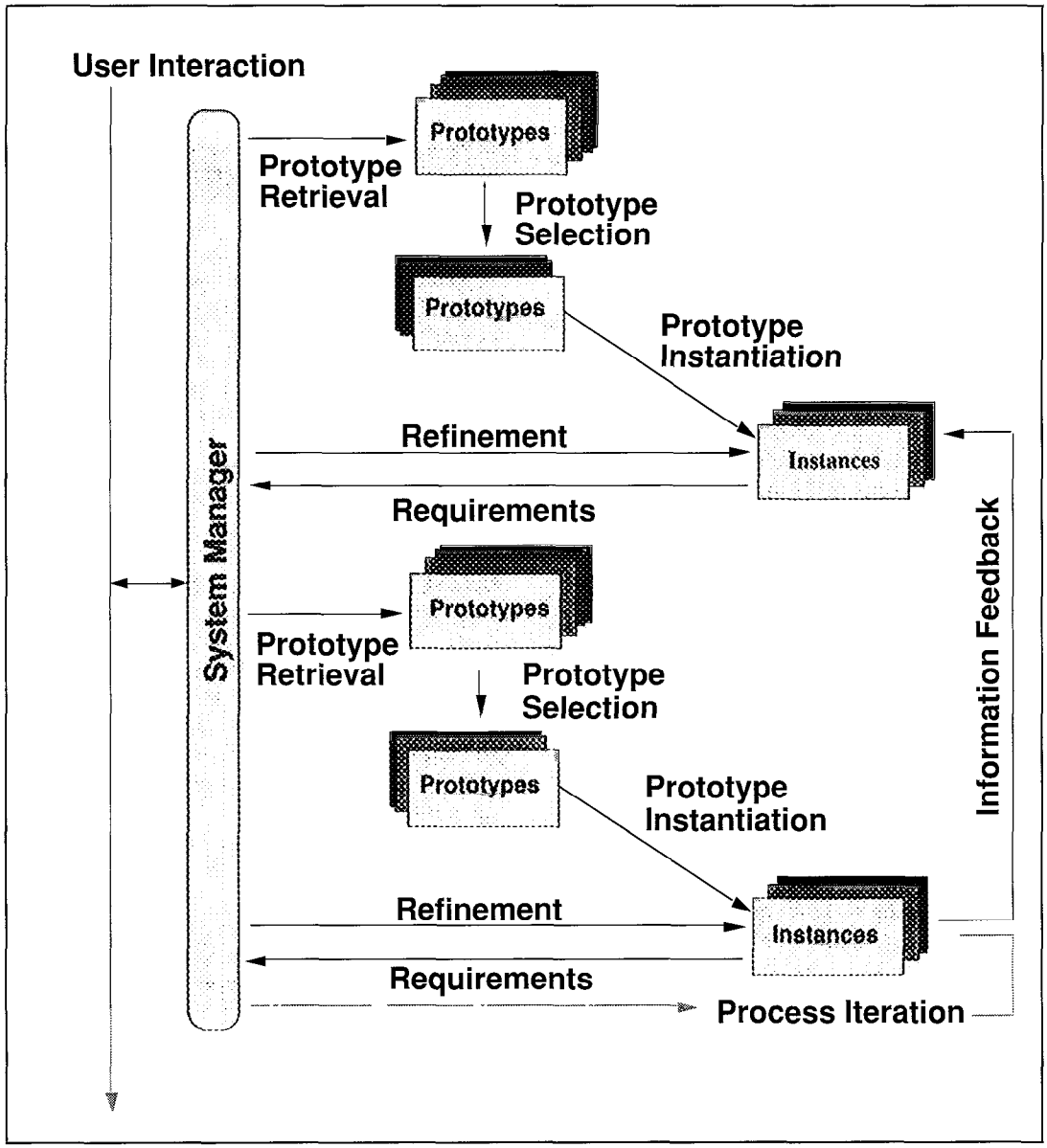


Figure 8. Routine Design by Design Prototype–Instance Refinement.

joint with the original prototype. On rare occasions, a design prototype is generated without precedent, for example, the design of the airplane, although, even here, a well-defined process could be used to explain its generation.

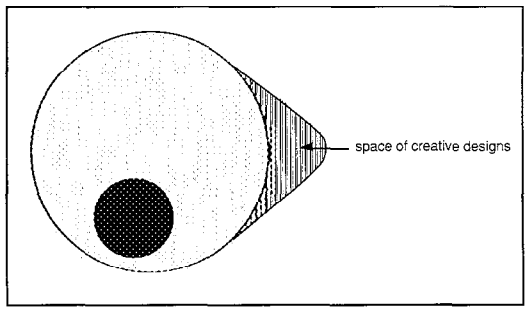


Figure 7. State Space of Creative Designs.

In creative design, the role of context and the designer's perception of it play an increasingly important part. Because design is being viewed as a process here, it could be argued that all new variables are already implicit in the processes to be used. To counter such arguments, it is suggested that designers work, of necessity, within a well-defined context of their choosing. The context is defined by the available design prototypes. However, there comes a time during the design process when the designer decides that s/he wishes to move outside the available design prototypes

to find new variables. This move can be seen as the designer changing the context in which s/he is working. The four main computational processes that appear capable of producing the new variables needed for creative design are combination, analogy, mutation, and first principles (these processes are not discussed here, however).

Discussion

Design requires a representation framework that has sufficient expressive power to capture the nature of the concepts that support design processes. The use of a knowledge representation schema such as design prototypes provides such a framework. It separates the knowledge from the computational processes that operate on it. The use of this representation effectively provides a translator between structure, which can be seen as the syntax of a design, and function, which can be treated as the semantics of a design. Such an articulation is useful not only in the production of designs but also in their analysis and evaluation. For example, traditional computer-aided design systems produce a design description in their databases that maps onto the syntax of a design. For these databases to be useful for other than purposes of graphic representation, a translation to the semantics of the design is needed.

It is tempting to view the design prototype schema as producing a rigid transformation between function and structure that is incapable of providing the basis for anything more than parameterized design. Certainly, it can be argued that this schema readily supports the notion of design fixation, where the provision of a design description for a specified set of functions limits the designer's ability to produce structures other than those found in the design description. However, the delineation of function from structure and their connection through behavior breaks the function-structure nexus and still maintains the association derived from experience.

The design prototype representation schema aims to match the expectations of a designer who utilizes computational processes in the production of a design. It readily provides a framework that supports both routine and nonroutine design processes.

Acknowledgments

I would like to thank the reviewers who assisted in refining this article. The work described here has been supported by the Australian Research Council, the U.S. National Science Foundation, and the United Kingdom Science and Engineering Research Council.

References

Bobrow, D. G. 1984. Qualitative Reasoning about Physical Systems: An Introduction *Artificial Intelligence* 24(1–3): 1–5.

Brown, D. C., and Chandrasekaran, B 1985 Expert Systems for a Class of Mechanical Design Activity In *Knowledge Engineering in Computer-Aided Design*, ed. J S. Gero, 259–282 Amsterdam: North-Holland

Coyne, R D ; Rosenman, M. A.; Radford, A D ; and Gero, J. S. 1987 Innovation and Creativity in Knowledge-Based CAD In *Expert Systems in Computer-Aided Design*, ed. J. S. Gero, 435–465 Amsterdam: North-Holland.

Coyne, R D.; Rosenman, M. A ; Radford, A. D.; Balachandran, M ; and Gero, J. S. 1990. *Knowledge-Based Design Systems*. Reading, Mass.: Addison-Wesley.

Durand, J-N-L 1802 *Precis des Legons d'Architecture* Paris: Ecole Polytechnique

Gero, J S. 1987. Prototypes: A New Schema for Knowledge-Based Design, Working Paper, Architectural Computing Unit, Univ of Sydney.

Jones, J C., and Thornley, D., eds. 1963 *Conference on Design Methods*. Oxford: Pergamon

Lawson, B. R. 1980. *How Architects Think*. London: Architecture Press

Osherson, D. N., and Smith, E. E 1981 On the Adequacy of Prototype Theory as a Theory of Concepts *Cognition* 9(1): 35–58

Schank, R. C., and Abelson, R. 1975 Scripts, Plans, and Knowledge. In Proceedings of the Fourth International Joint Conference on Artificial Intelligence, 151–157 Menlo Park, Calif.: American Association for Artificial Intelligence.

Simon, H A 1969. *The Sciences of the Artificial*. Cambridge: MIT Press

Tham, K W ; Lee, H. S.; and Gero, J S. 1990 Building Envelope Design Using Design Prototypes. In *AI in Building Design: Progress and Promise, ASHRAE Symposium* Forthcoming.



John Gero is professor of design science, director of the Design Computing Unit, and director of the Key Centre of Design Quality at the University of Sydney. He is also an adjunct professor at Carnegie-Mellon University. He is the author-editor of 16 books and 200 research papers He has been a visiting professor of architecture, civil engineering, computer science, and mechanical engineering at universities in the United States, United Kingdom, and France. His current research interests include computational processes that support creativity.