DESIGN PROTOTYPES: A KNOWLEDGE REPRESENTATION SCHEMA FOR DESIGN

JOHN S. GERO

Design Computing Unit
University of Sydney
NSW 2006 Australia

Abstract. This article commences 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.

DESIGN

Although there are designers who claim design is a mysterious activity not amenable to scientific examination, there continues to be research into design. Although there are publications by designers on how to design dating back to Roman times, notably by Vitruvius, the nineteenth century design thinkers commenced work on articulating design as a process (Durand, 1802). However, it was not until the 1960s that major research programs were initiated. These were originally founded on the systems view and utilised concepts from operations research (Jones and Thornley, 1963). More recently, information processing models founded on artificial intelligence concepts have provided an impetus for renewed research into design in its various aspects (Simon, 1969; Coyne et al., 1990). Many foundational ideas in artificial intelligence 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 4,000 years. Approximately 2,000 BC, Hammurabi, King of Babylon, enacted a law which both recognized design and made it dangerous, figure 1. Design research has a number of goals including: gaining a better understanding of design, developing tools to aid human designers, and the potential automation of some design tasks.

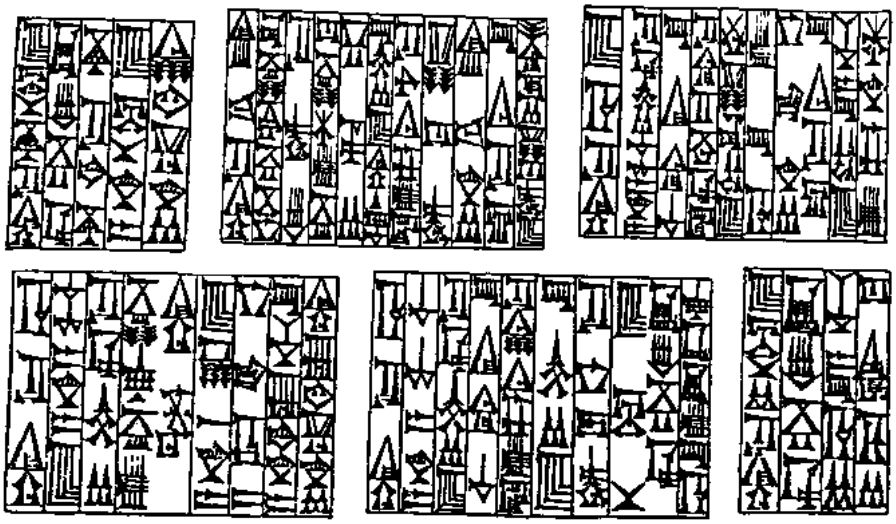


Figure 1. *Hammurabi's Code, from an engraving on a stella in cuneiform in the Louvre, Paris. "If a designer/builder has designed/built a house for a man and his work is not good, and if the house he has designed/built falls in and kills the householder, that designer/builder shall be slain."*

Design appears to be carried out differently to 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 commences 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 may 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 artefacts. They do this by positing functions to be achieved and producing descriptions of artefacts capable of generating those 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 meta-goal of design is to transform requirements, more generally termed *functions* which embody the expectations of the purposes of the resulting artefact, into design descriptions.

The result of the activity of designing is a design description. This design description generally is represented graphically, numerically, and/or textually. The purpose of such a description is to transfer sufficient information about the designed artefact 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 taken about what values should be taken by those variables. The activity of designing is carried out with the expectation that the designed artefact will operate in the natural world and the social world. These 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 equally importantly by additional necessary features. Designing involves exploration: exploring what variables may 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; 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 is the context 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 which operates within a context which 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 artefact being described is capable of producing those functions. For example, when designing windows some of the functions include the provision of daylight, control of ventilation and the provision of access to a view. The design description would take the form of drawings and notes. Thus, a naive model of design is

F -> D

where -> is some transformation. There is, however, no direct transformation capable of achieving this result.

A design description represents the artefact's elements and their relationships, this is labelled *structure*, S. In the window design example the artefact'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, i.e.

S -> D

Another model of design is

F ->S

Here a transformation occasionally does exist in the form of a direct mapping between function and structure, often termed *catalog lookup*. This occurs at the element level of an artefact and is not considered designing. More generally, no direct transformation between function and structure exists. This 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 may be viewed in two ways. There is the behavior of the structure, *B*s (where *B*s is a set), which is directly derivable from structure

S -> *Bs*

In the window design example the behaviours of the structure include the light flux transmitted, the ventilation rate, the various solar gains, etc. 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 behaviours for the window design example include light transmission, ventilation rates, solar collection, etc. The expected behavior provides the syntax by which the semantics represented by function can be achieved

F -> *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 <-> *B*s

where <□> is a comparison. This comparison process is termed *evaluation* in design.

Another model of design is

F -> *B*e

*B*e-> 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 that structure. This process is that of *synthesis.*

When structures are synthesized they produce their own behaviors which may be a useful superset of the expected behaviors. This may change the 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 structure. This leads to a change in expected behavior.

Figure 2 shows how these transformations appear in design.

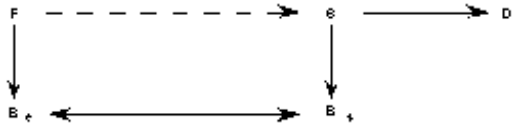


Figure 2. *Model of design as a process*
*B*e = set of expected behaviors
*B*s = set of actual behaviors
D = design description
F = set of functions
S = structure
-> = transformation
--> = occasional transformation
<-> = comparison

A general model of design as a process involves the following activities, figure 3,

- formulation
- synthesis
- analysis
- evaluation
- reformulation
- production of design description

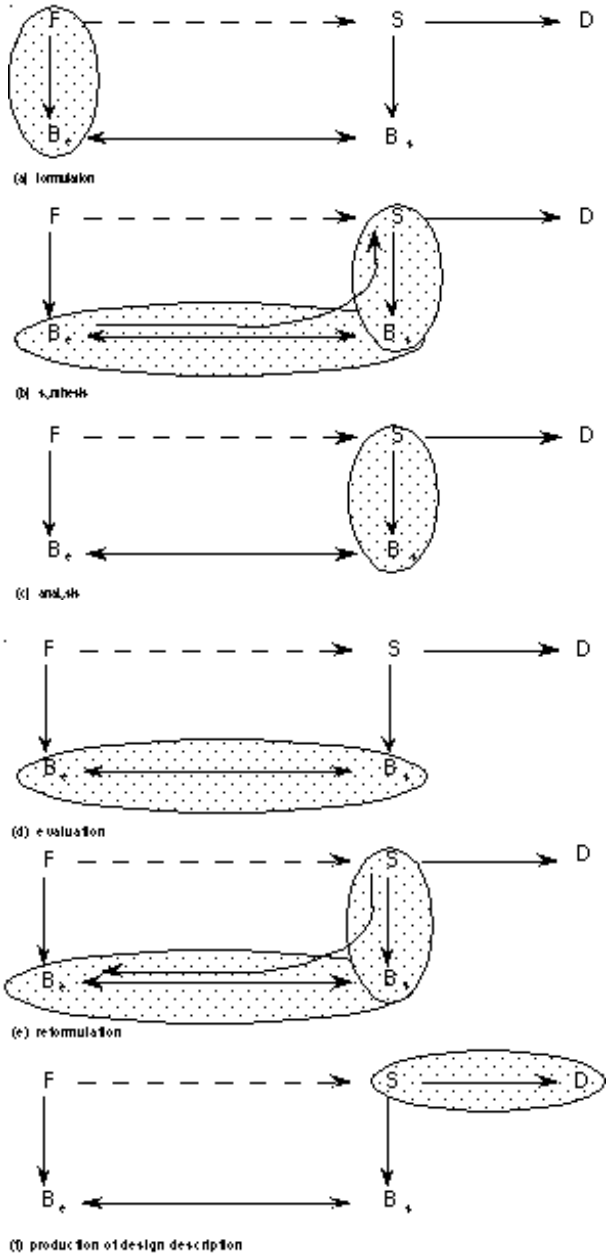


Figure 3. *Activities in design*
(a) formulation, (b) synthesis, (c) analysis, (d) evaluation (e) reformulation, (f) production of design description

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

THE CONCEPTUAL SCHEMA *DESIGN PROTOTYPES*

SCHEMAS

How is it that designers can commence designing with incomplete information and before all the relevant information is available? Indeed, since 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 analyse for?

It is suggested that human designers form their individual design experiences into generalised 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 like design cases and form a class from which individuals may 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 those components. It will be seen later that more than these needs to be incorporated within a schema. There appear to be no satisfactory schemas available for the representation of this generalized design knowledge which have sufficient expressiveness and power. There are none which can be used to explain how a design commences 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 very rich, young bachelor's car. In both cases there can be no substitute. Each archetype may provide an analogue 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, i.e. the process of reproduction, belies the design endeavor. Prototypes are the first on which others are modeled. This is the basis 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 grouping of elements, derived from like design cases, which provides the basis for the commencement and continuation of a design. Design prototypes do this by bringing together in one schema all the requisite knowledge appropriate to that design situation.

A designed artefact may be broadly interpreted in terms of the three variable groups of function, structure and behavior. The level of specificity in each of these depends on the granularity and level of abstraction being represented. Thus, at an early stage of designing an appropriate design prototype may contain primarily function and behavior with little information on structure. Whilst at a later time an appropriate design prototype will 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 that 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 time sequence bound.

Although it is well recognized that there is no function in structure and vice-versa that there is no structure in function, human and design experience produces a connection between function and structure. Once that connection is learned it is very difficult to unlearn. Once the connection between behavior and structure is made and the connection between behavior and function is made it forms the basis of much of a designer's knowledge. It is function, structure, behavior *and* relationships which form the foundation of the knowledge which must be represented in order 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 function. For example, the label of a particular copier 'Xerox' is slowly taking on the meaning of its function, i.e. 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, behavior categories and can take the form of a dependency network. Relational knowledge identifies the relevant variables in going from function to behavior, from behavior to structure and in the inverse direction. Relational knowledge allows for the specialization of the information in a prototype to a specific design situation (see section '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 amongst the variables. Computational knowledge is used to determine values of variables.

Constraints appear in both the 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, i.e. from the context (C).

In addition there is knowledge concerning the design prototype itself (*K*p). This comprises the *typology* (T) of the design prototype which identifies the broad class of which the design prototype is a member, while *partitions* (P) represent the subdivisions of the concept represented by the prototype. Partitioning a design prototype supports viewing it from many perspectives. Once the

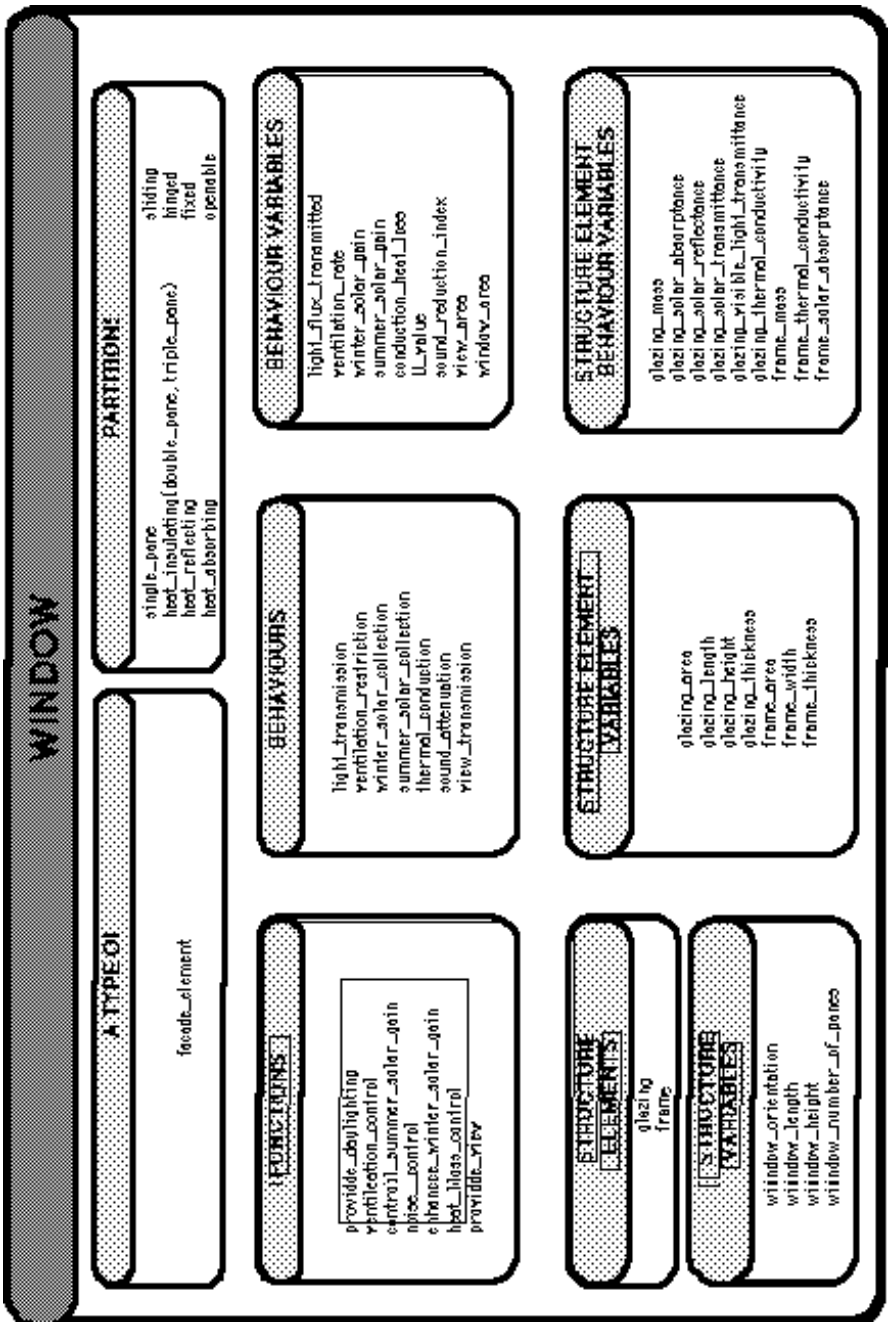
partition or combination of partitions is selected only information pertaining to these partitions will be made available. In this sense, partitioning of design prototypes ultimately reduces the space of potential designs.

A design prototype, P, may be represented symbolically as

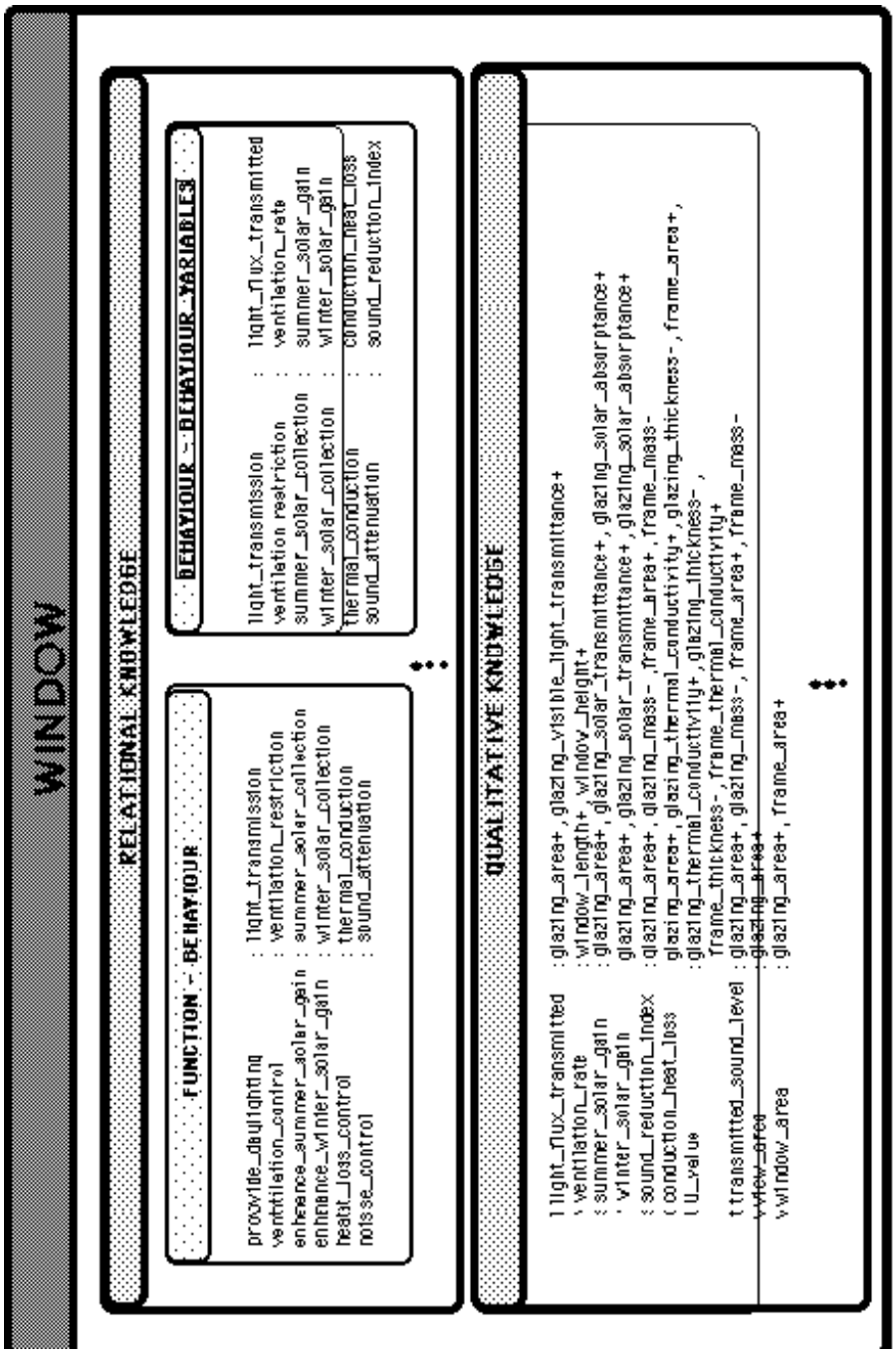
$$P = (F, B, S, D, K, C)$$

where $B = (Bs, Be)$
C = context
$K = (Kr, Kq, Kc, Kct, Kp)$
$Kp = (T, P)$
P = partition
T = typology

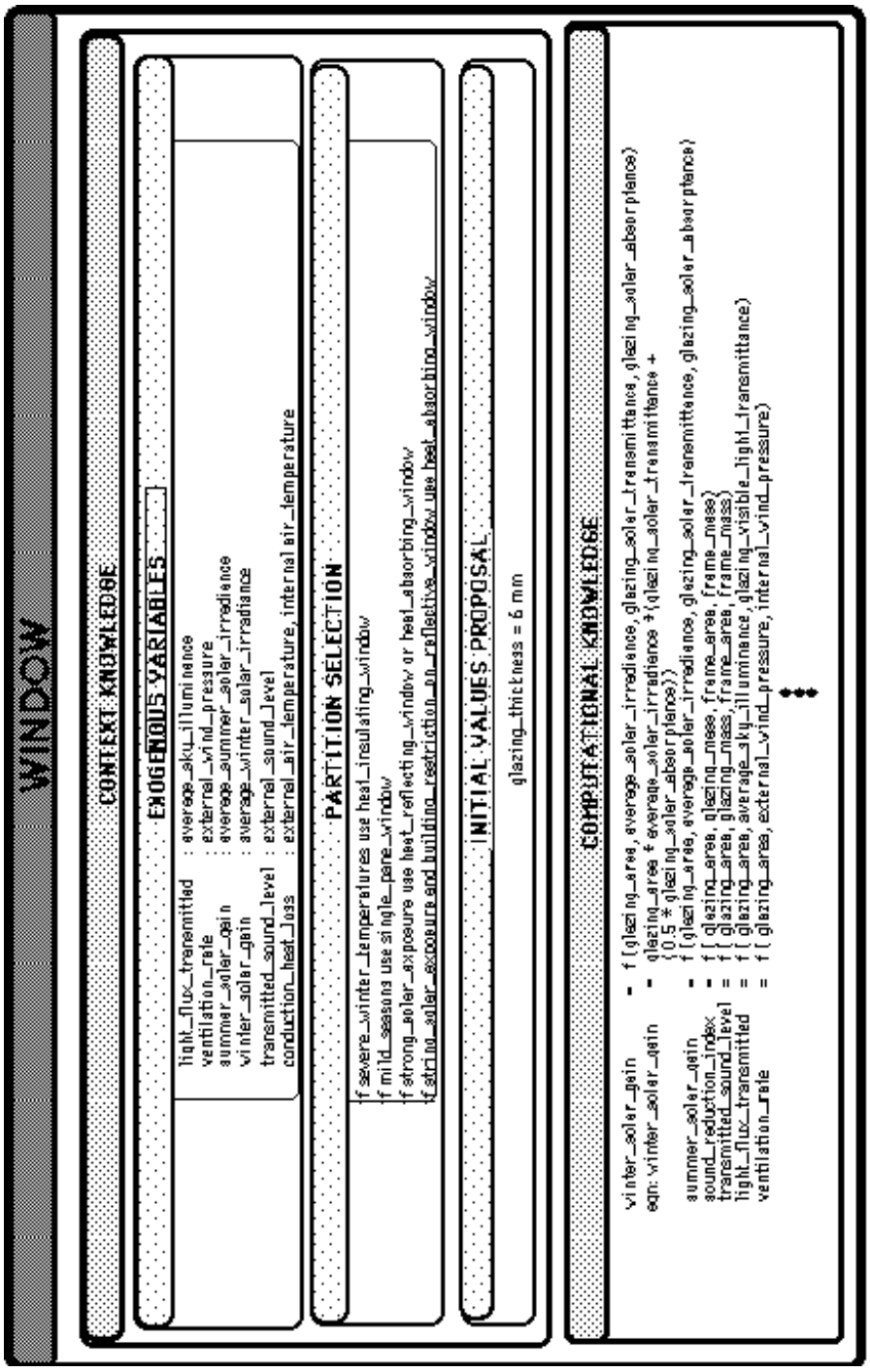
Figure 4 shows a diagrammatic representation of the design prototype schema for the knowledge associated with window design.



(a)


WINDOW
RELATIONAL KNOWLEDGE
FUNCTION - BEHAVIOUR
provide_daylighting : light_transmission
ventilation_control : ventilation_restriction
enhance_summer_solar_gain : summer_solar_collection
enhance_winter_solar_gain : winter_solar_collection
heat_loss_control : thermal_conduction
noise_control : sound_attenuation
BEHAVIOUR - BEHAVIOUR_VARIABLES
light_transmission : light_flux_transmitted
ventilation restriction : ventilation_rate
summer_solar_collection : summer_solar_gain
winter_solar_collection : winter_solar_gain
thermal_conduction : conduction_heat_loss
sound_attenuation : sound_reduction_index
QUALITATIVE KNOWLEDGE
llight_flux_transmitted : glazing_area+, glazing_visible_light_transmittance+
vventilation_rate : window_length+ window_height+
ssummer_solar_gain : glazing_area+, glazing_solar_transmittance+, glazing_solar_absorptance+
'winter_solar_gain : glazing_area+, glazing_solar_transmittance+, glazing_solar_absorptance+
ssound_reduction_index : glazing_area+, glazing_mass- ,frame_area+, frame_mass-
cconduction_heat_loss : glazing_area+, glazing_thermal_conductivity+, glazing_thickness- , frame_area+,
UU_value : glazing_thermal_conductivity+ , glazing_thickness- ,
frame_thickness- ,frame_thermal_conductivity+
ttransmitted_sound_level : glazing_area+ , glazing_mass- , frame_area+ , frame_mass-
vview_area : glazing_area+
vwindow_area : glazing_area+ , frame_area+

(b)



(c)

Figure 4. *Diagrammatic representation of the design prototype schema for the knowledge associated with window design (Tham et al., 1990).*

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 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, building proposal codes, and so on. A designer will draw on many design prototypes during the course of developing any design.

DESIGNING USING DESIGN PROTOTYPES

Designing using design prototypes may 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 very early in a design (Lawson, 1980). Later

in this paper the specifics of routine and non-routine design are presented. Here, a descriptive outline of designing using design prototypes is given.

A designer commences with required functions from a client. Sometimes clients also specify required structures. These are thought of as requirements which are used to retrieve potentially useful design prototypes on the basis that they are indexed via these requirements. These retrieved design prototypes represent the set of concepts that a designer 'remembers' when he or she 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 which may 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 of the knowledge in the design prototype may be useful in the particular context so it is pruned and that pruning propagated using the dependency knowledge.

The pruned instance is now the equivalent of a formulated design problem at that 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 those variables are transformed into requirements. These are then used to retrieve additional design prototypes which have the potential through their instances to produce those 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 commencement of a design at any level of available information. Since 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. Whilst retrieval by structure alone introduces functions associated with that structure. Design prototypes encode what are the appropriate behaviors to analyse for. Design prototypes provide a knowledge representation schema separate from the specific computational processes. Computational processes which support design are described elsewhere (Coyne et al., 1990). Design prototypes readily provide a framework which supports both routine and non-routine design which are the subjects of the next two sections.

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 the following have proved to be useful.

Routine design may be defined as that design which 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 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.

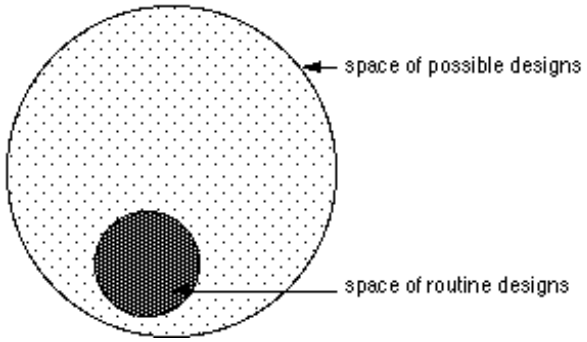


Figure 5. *State space of routine designs*

Innovative design may be defined as that design which 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 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.

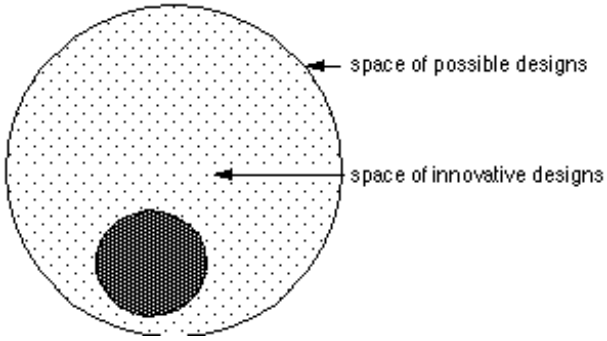


Figure 6. *State space of innovative designs*

Creative design may be defined as that design which 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.

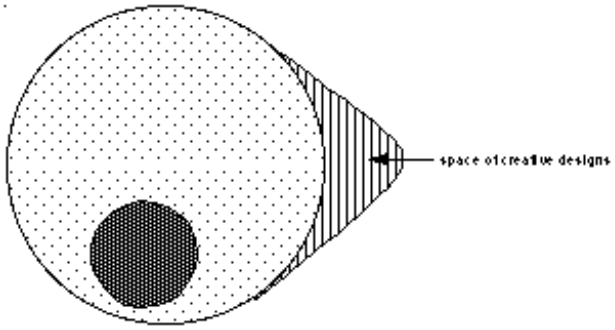


Figure 7. *State space of creative designs*

ROUTINE DESIGN

Routine design may be viewed as *design prototype/instance refinement*. Design prototypes are retrieved, selected and instances produced. These instances are refined in two ways. The first is by pruning the set of variables to the applicable set through a specification of applicable functions and/or structures and/or behaviors and propagating that specification. The second 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 the qualitative knowledge and computational knowledge are separated from the representation of the design prototypes and from any system architecture. The processes cover:

- formulation
- synthesis
- analysis
- evaluation
- reformulation
- production of design description

The use of a conceptual schema, such as design prototypes, which collects together all the requisite knowledge, provides a basis for routine design.

INNOVATIVE DESIGN

Innovative design may be viewed as design prototype/instance refinement with an adaptation of some of the knowledge concerning applicable ranges of variable values, i.e. *design prototype/instance adaptation*. This requires additional processes for adaptation and then use of dependency knowledge to assist in the confirmation of the utility of any change.

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, i.e. *design prototype generation*. In most cases new prototypes are produced from old by changing them. It is possible to adapt a design prototype sufficiently so that the new design prototype is disjoint with the original prototype. On rare occasions a design prototype is generated *de novo*, for example, the design of the aeroplane. Although even here a well-defined process could be used to explain its generation.

In creative design the role of context and the designer's perception of it play an increasingly important part. Since 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. In order 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 he or she wishes to move outside the available design prototypes in order to find new variables. This can be seen as the designer changing the context in which he or she is working. The three main computational processes which appear capable of producing the new variables needed for creative design are analogy, mutation and first principles. These will not be discussed here, however.

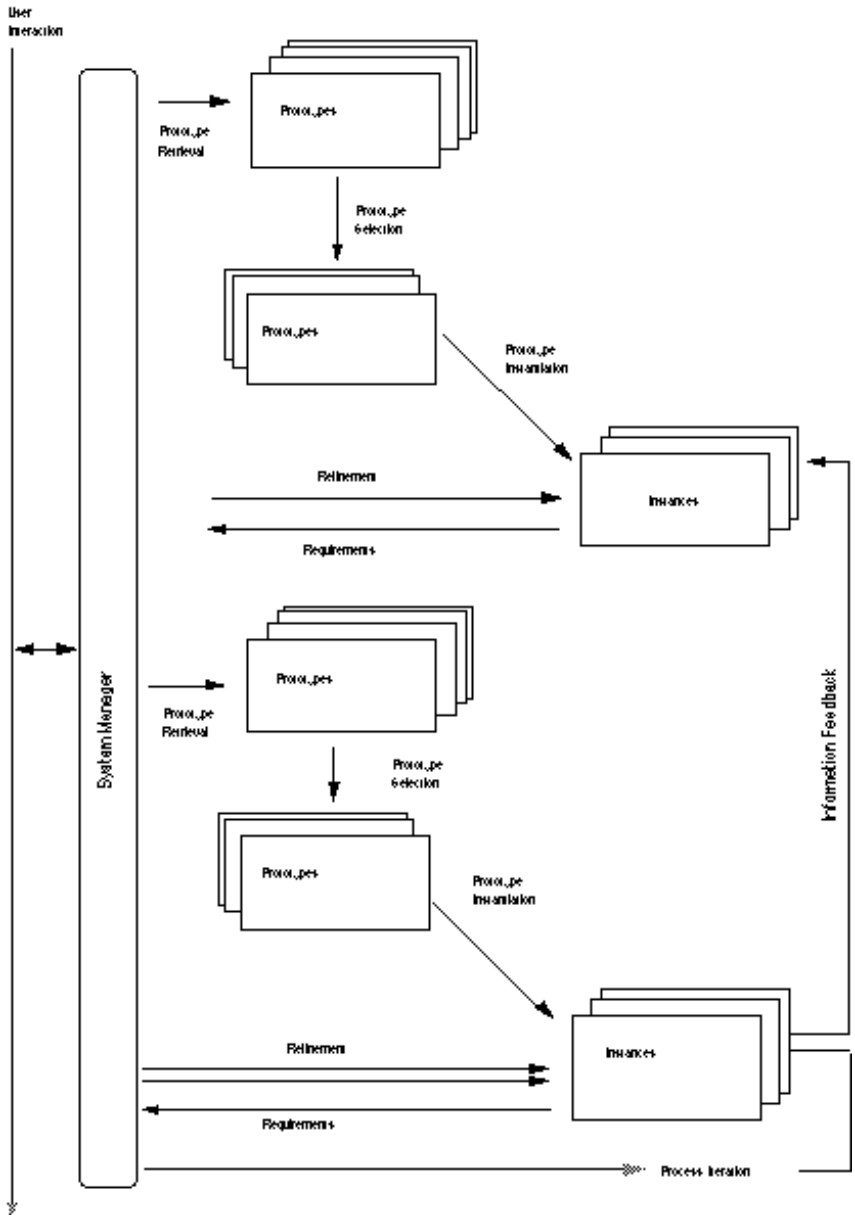


Figure 8. *Routine design by design prototype/instance refinement*

DISCUSSION

Design requires a representation framework which has sufficient expressive power to capture the nature of the concepts which support design processes. The use of a knowledge representation schema such as design prototypes allows for this. It separates the knowledge from the computational processes which operate on it. The use of this representation effectively provides a translator

between structure, which may be seen as the syntax of a design, and function, which may 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 CAD systems produce a design description in their databases which map onto the syntax of a design. For these databases to be useful for other than graphical representation purposes 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 which 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. Design fixation is 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 whilst still maintaining the association derived from experience.

The design prototype representation schema aims to match the expectations of a designer who utilises computational processes in the production of a design. It readily provides a framework which supports both routine and non-routine design processes.

ACKNOWLEDGEMENTS

I would like to thank the reviewers who assisted in refining the paper. The work described here has been supported by the Australian Research Council, the US-National Science Foundation and the UK-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, *Knowledge Engineering in Computer-Aided Design*, ed. J. S. Gero, Amsterdam, North-Holland, pp.259-282.

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

Coyne, R. D.; Rosenman, M. A.; Radford, A. D.; and Gero, J. S. 1987. Innovation and creativity in knowledge-based CAD, *Expert Systems in Computer-Aided Design*, ed. J. S. Gero, Amsterdam, North-Holland, pp. 435-465.

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

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. *IJCAI-75*, Georgia, USSR: Tsibilisi, pp.151-157.

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

Tham, K. W.; Lee, H. S.; and Gero, J. S. 1990. Building envelope design using design prototypes, *AI in Building Design: Progress and Promise*, ASHRAE Symposium, St Louis, Missouri, USA (to appear).