ELSEVIER

Knowledge-Based Systems 9 (1996) 435–448

Knowledge-Based SYSTEMS

Creativity, emergence and evolution in design

John S. Gero

Key Centre of Design Computing, Department of Architectural and Design Science, University of Sydney, NSW 2006 Australia

Received 19 May 1995; revised 18 July 1996; accepted 23 July 1996

Abstract

This paper commences by outlining notions of creativity before examining the role of emergence in creative design. Various process models of emergence are presented; these are based on notions of additive and substitutive variables resulting in additive and substitutive schemas. Frameworks for both representation and process for a computational model of creative design are presented. The representational framework is based on design prototypes whilst the process framework is based on an evolutionary model. The computational model brings both representation and process together.

Keywords: Creative design; Emergence; Evolutionary processes; Design computing

1. Creativity in design

In order to develop and describe any model of creativity in design we need to have an acceptable conception of design. Design, in one sense, can be conceived of as a purposeful, constrained, decision making, exploration and learning activity. Decision making implies a set of variables, the values of which have to be decided. Search is the common process used in decision making. Exploration here is akin to changing the problem spaces within which decision making occurs. Learning implies a restructuring of knowledge. The designer operates within a context which partially depends on the designer's perceptions of purposes, constraints and related contexts. These perceptions change as the designer explores the emerging relationships between putative designs and the context and as the designer learns more about possible designs. Whilst much more can be said about design [1–5], this provides a sufficient conception to provide a context within which the rest of this paper sits.

Creativity and creativity in design, in particular, have many interpretations [6–11]. There is a clear distinction to be drawn between considering creativity as residing only in the artefact and evaluated by society and considering that certain processes have the potential to produce artefacts which may be evaluated as creative. This paper adopts the approach that whilst the creativeness of an artefact is societal in its evaluation there may be processes which can aid in the understanding of how creative artefacts may be produced.

Creativity, it has been suggested, is not simply concerned with the introduction of something new into a design, although that appears to be a necessary condition for any process that claims to be labelled creative. Rather, the introduction of 'something new' should lead to a result that is unexpected (as well as being valuable). More formally we can describe routine designing as following a defined schema where the expectations of what follows is defined by the schema. Creative designing, which is part of non-routine designing, can be described as perturbing the scheme to produce unexpected and incongruous results. These new results are still understandable either in a current or shifted context.

Although the boundary between routine and creative designs is difficult to define there is less difficulty in articulating differences between processes used in the production of routine and creative designs. This paper elaborates a process-oriented view of computational design creativity. It uses the notions of unexpectedness and emergence within a schema-based view of design.

2. Model of creative design

One useful way to provide a framework for design is through the conceptual schema design prototypes [12]

Email: john@arch.usyd.edu.au

PII S0950-7051(96)01054-4

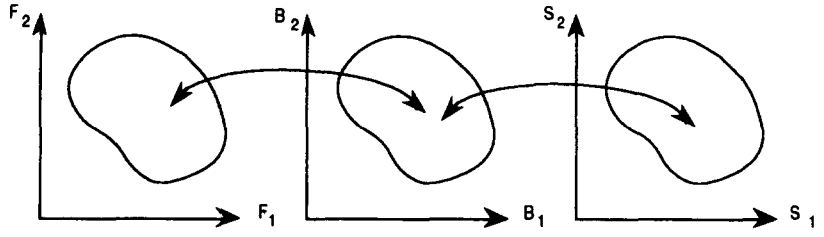

Fig. 1. The three subspaces of function (F), behaviour (B) and structure (S) which constitute the state space of designs, plus the locus of the transformations between them.

which articulates a function–behaviour–structure + knowledge framework. Thus, the state space representation of designs has three subspaces or abstractions: the structure space, S (often called the decision space); the behaviour space, B (often called the performance space); and the function space, F (which defines the artefact's teleology). Fig. 1 shows these three subspaces which constitute the state space of designs.

Whilst there are transformations which map function to behaviour and vice-versa and structure to behaviour and vice-versa, there are no transformations which map function to structure. This is a version of the no-function-in-structure principle [12, 13] where the teleology of an artefact is not found in its structure but is a contextual interpretation of its behaviour. The corollary: no-structure-in-function also holds. This may, at first glance, be counter-intuitive. The reason is that in human experience once a phenomenological connection between function and structure is made it is hard to unmake it.

Often only the structure and behaviour spaces are considered in computational models although function provides an important articulation of ideas about design. Typical computational models of design can be grouped under such processes as simulation, optimization, generation, decomposition, constraint satisfaction, and more generally search and exploration. All of these share one concept in common, namely that structures are produced in a design process and their resultant behaviours are evaluated. It is only recently that the function of the artefact being designed is beginning to be brought into the computational model [14–17].

2.1. Creativity and humour

Creativity is involved with the production of an unexpected result through the confluence of two schemas. The first schema provides a set of routine expectations, the second schema is needed to understand the unexpected

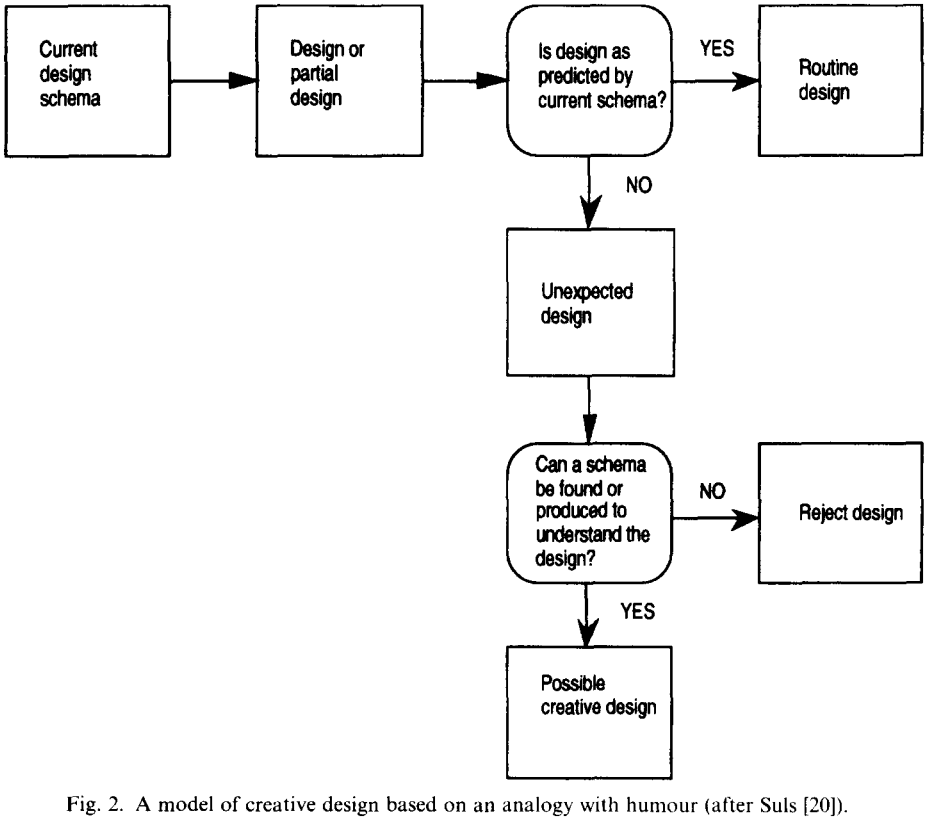

Fig. 2. A model of creative design based on an analogy with humour (after Suls [20]).

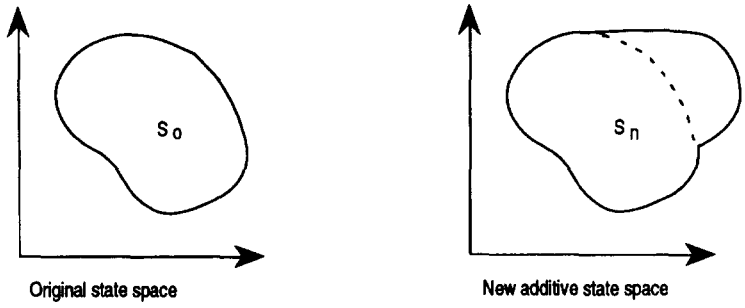

Fig. 3. The additive state space view.

result. The unexpected result can be produced in a number of different ways described later in this paper.

A model for creative design can be found by analogy to models of humour. Humour "... arises from the view of two or more inconsistent, unsuitable, or incongruous parts or circumstances, considered as united in [a] complex object or assemblage, or as acquiring a sort of mutual relation from the peculiar manner in which the mind takes notice of them" [18]. Koestler [19] suggests that there is a continuity of creative insights in humour with those in science and poetry. "The logical pattern of the creative process is the same in all three cases: it consists of discovery of hidden similarities" [19].

Here is an example of the two schema paradigm of humour: An unskilled man, desperate for work, turns up at a construction site and asks the foreman if there are any jobs available. The foreman thinks he looks unintelligent, and doesn't believe he has the qualifications or knowledge for a job but, being a compassionate person, decides to give him a chance. He says "I'll give you a job if you can tell me the difference between girder and joist." The man scratches his head and says, "Easy! Can't be caught out by that one. Everyone knows the difference ... Goethe wrote *Faust* and Joyce wrote *Ulysses*." Here the response introduces new variables which require a new schema to understand them.

A model of this paradigm in design terms is presented in Fig. 2.

2.2. State space representation of creative design

For a given set of variables and processes operating within a bounded context or focus any computational model will construct a bounded (although in some cases countably infinite) state space. Creative design can be represented in such a state space by a change in the state space. Any of the subspaces in Fig. 1 for function, behaviour or structure could be changed although, in general, in design it is the structure space that is changed. There are two classes of change possible: addition and substitution. This is based on Stevens' two forms of psychological representational scales [21]. The additive view is presented conceptually in Fig. 3 where the new state space S_n totally contains the original state space S_o, i.e. $S_o \subset S_n$ and $S_n - S_o \neq \varnothing$.

The implication of the additive view is that variables are added to the existing stock of variables. Gero and Kumar [22] have demonstrated how the addition of structure variables allows design spaces that contain infeasible behaviour spaces to be made feasible. Further, they demonstrated how the addition of structure variables can improve the behaviour of an already optimized design.

The substitution view is presented conceptually in Fig. 4 where the new state space S_n does not cover the original state space S_o, i.e. $S_o \not\subset S_n$.

The implication of this substitutive view is that some existing variables are deleted and others added. There is no nexus between the number of existing variables deleted and the number of new variables added. As will be seen later this view matches the concept of emergence. The concepts of additive and substitutive spaces also apply to schemas as will be discussed in Section 4.

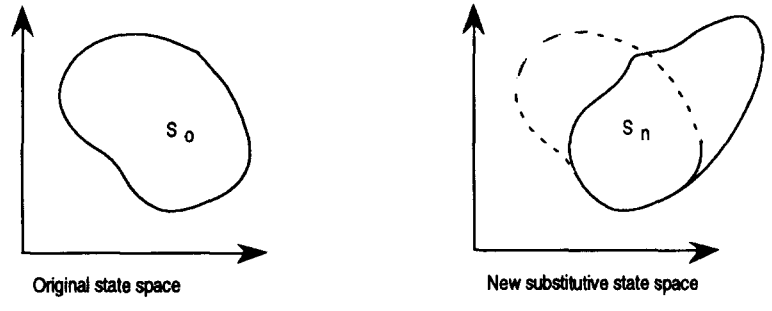

Fig. 4. The substitutive state space view.

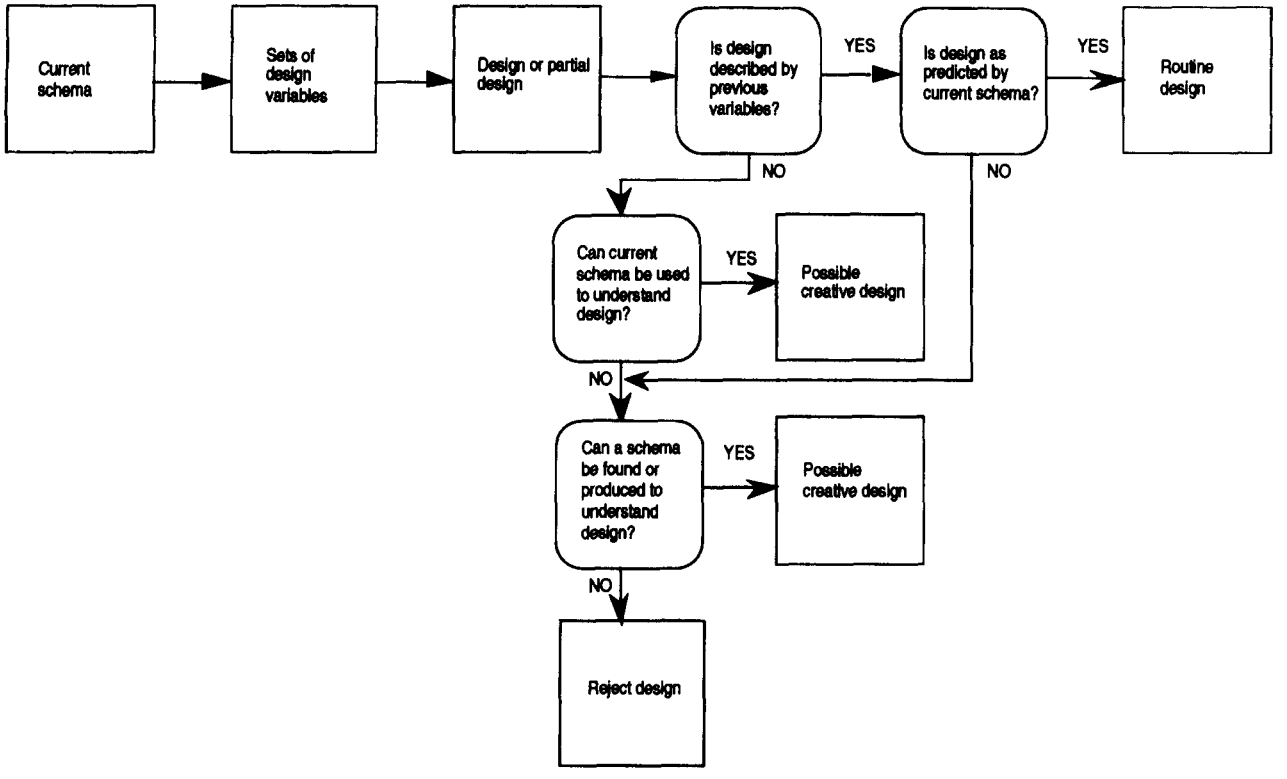

Fig. 5. Diagrammatic representation of the new variable/new schema model of creative design developed by expanding the model in Fig. 2 to include variables as well as schemas.

2.3. *Model of creative design*

The model of creative design implicit in the previous two sections contains two concepts:

1. introduction of new variables
2. introduction of new schema

These two concepts interact with each other. The model outlined in Fig. 2 can be elaborated to include these concepts as in Fig. 5.

3. Emergence

A property that is only implicit, i.e. not represented explicitly, is said to be an emergent property if it can be made explicit. Emergence is considered to play an important role in the introduction of new schemas and consequently new variables. Emergence is a recognised phenomenon in visual representations of structure. It maps directly onto the concept of changing schemas since a new schema is generally needed to describe the emergent property. Consider the case of the three equilateral triangles shown in Fig. 6(a).

If the schema is concerned with triangles then only triangles will be found. However, another schema for the structure will find the trapezoid in Fig. 6(b) which was not explicitly represented in Fig. 6(a). A more striking example of visual emergence can be found in Fig. 7. Consider the object in Fig. 7(a). It is copied into three different locations as shown in Fig. 7(b). Human observers can readily see the 'phantom' forms of the star-of-David and various triangles. In order to see these, new schemas are needed and a computational model of emergence must be able to utilise this concept [23].

Emergence is not limited, however, only to structure. Emergence can also apply to behaviour and function. Finke [24] gives examples of emergent function for a given fixed structure (presumably determined by reasoning about the possible behaviours of the structure and about possible teleologies associated with those behaviours), Figs. 8 and 9.

There is remarkably little on emergence and computation generally although there is some work [25–27]. Recently, there has been considerable research aimed at providing computational analogs of emergence in the spatial domain [28–30].

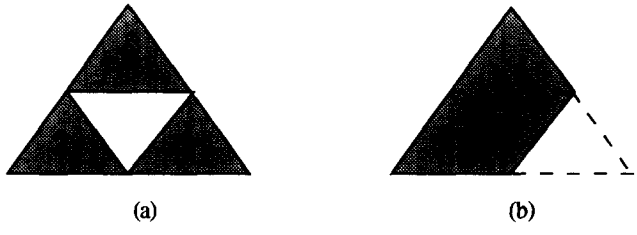

Fig. 6. (a) Three equilateral triangles, which are the only shapes explicitly represented. (b) One emergent form in the shape of a trapezoid moving that shape from being implicit to being explicit.

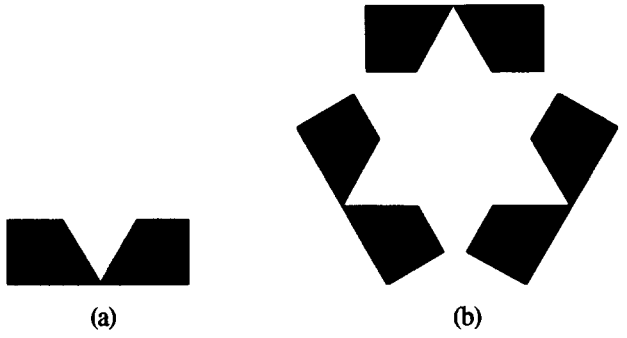

Fig. 7. (a) Single object. (b) Configuration of three copies of the object resulting in a number of emergent forms.

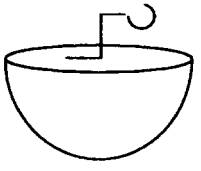

Fig. 8. Fixed structure used in function emergence [24].

4. Models of creative design processes

What kinds of processes are capable of modifying a design space in either an additive or a substitutive manner. Processes for the addition of variables have been developed to a much greater extent than those which substitute a new schema for the old.

Let us consider a schema to embrace the three variable classes of function, F, structure, S, and behaviour, B. These three classes are operated on by processes, K, which connect them. This is the design prototype schema when the variable classes and their constituent processes occur within a context, C. A design prototype, P, can be defined as:

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

All the variable classes of F, B, S and C are open to be modified as is K.

4.1. Process for the addition of variables and their effects on schemas

We need to distinguish two kinds of results from any process capable of adding variables. Such processes can add variables which are either: (i) homogeneous or (ii) heterogeneous.

Homogeneous variable addition occurs when the added variable is of the same kind as an existing variable and the existing knowledge (perhaps with minor alterations) can be used to integrate it into the current schema. Fig. 10 shows the dependency network of a design prototype, whilst Fig. 11 shows the same dependency network modified by the introduction of a new homogeneous variable which can utilise the existing knowledge structure. An example of this is given in Gero and Maher [7]. The implication of this is that the existing schema can continue to be employed.

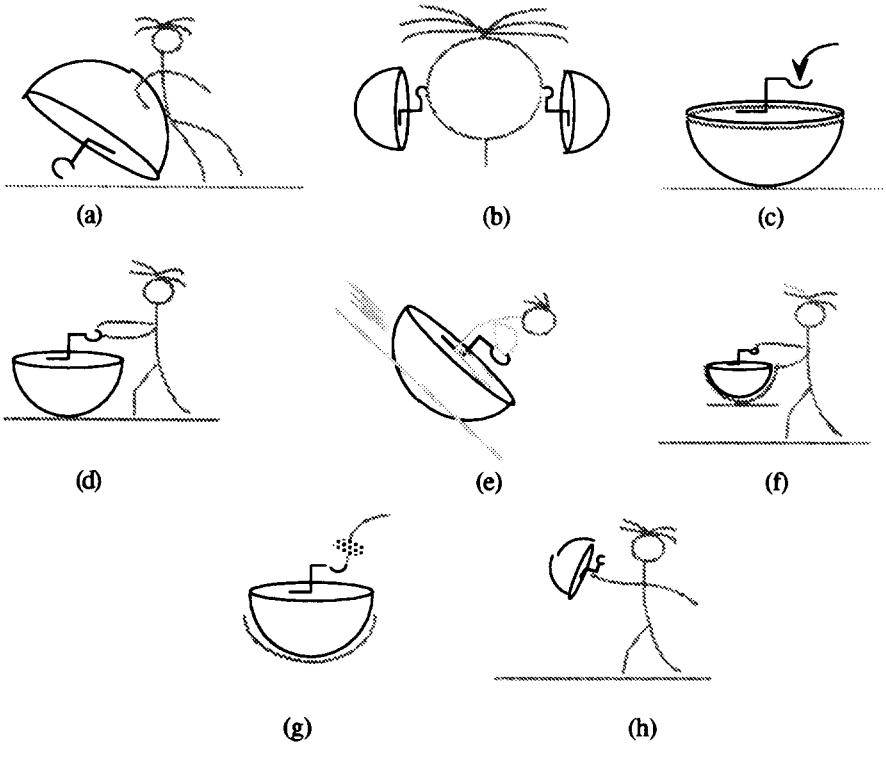

Fig. 9. Various functions the structure in Figure 8 could serve, such as (a) lawn lounge (furniture), (b) global earrings (jewellery), (c) water weigher (scientific instruments), (d) portable agitator (appliance), (e) snow sled (transportation), (f) rotating masher (tools and utensils), (g) top of spinner (toys and games) and (h) slasher basher (weapons) [24].

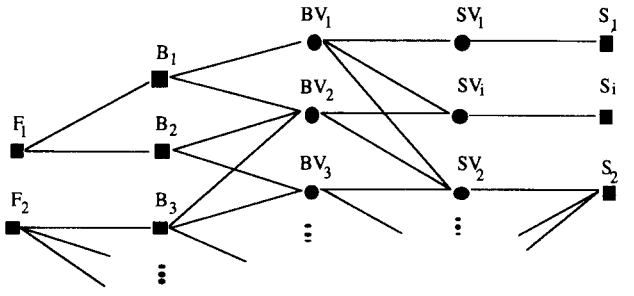

Fig. 10. Part of a dependency network in a design prototype showing the relationships among function, behaviour, behaviour variables, structure variables and structure (after [7]).

Heterogeneous variable addition occurs when the added variable is of a different kind than existing variables and the existing knowledge cannot be used to integrate it into the current schema. Fig. 12 shows the dependency network of Fig. 10 modified by the introduction of a heterogeneous variable. An example of this is given in [7].

Effects on schemas

Before proceeding with the description of processes for the addition of new variables it is appropriate to examine the two kinds of effects such additions can have on schemas. In a manner analogous to the effects on the state space there are two classes of schema effects possible: addition and substitution. The additive effect may occur when a homogeneous variable is added into the schema — it simply extends the existing schema without otherwise altering it, Fig. 13. The substitutive effect may occur when a heterogeneous variable is added into the schema provided the heterogeneous variable substitutes for one or more existing variables — it has the potential to change the schema being used, Fig. 14. This matches the concept of emergence.

What processes with their computational analogs exist to add variables? There appear to be a number of such processes of interest. Three will be described here:

1. combination
2. analogy
3. mutation

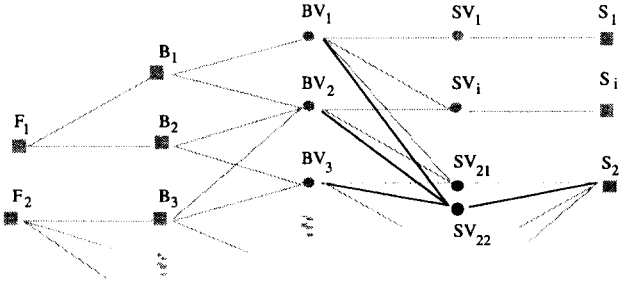

Fig. 11. Part of the dependency network shown in Fig. 10 modified by the addition of a homogeneous variable by dividing SV_2 into $\{SV_{21}, SV_{22}\}$ (after [7]).

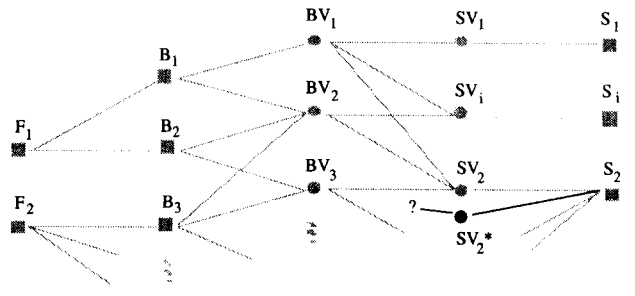

Fig. 12. Part of the dependency network shown in Fig. 10 modified by the addition of a heterogeneous variable by changing SV_2 into $\{SV_2, SV_2^*\}$ (after [7]).

Combination involves the addition of part or all of one design prototype called the combining design prototype to an existing design prototype called the focus design prototype. The variables which can be added can come from either structure behaviour or function.

Combination can be represented as:

$$F_{new} = F_f \wedge F_c$$

$$B_{new} = B_f \wedge B_c$$

$$S_{new} = S_f \wedge S_c$$

where the subscripts are:

new = combined design prototype

f = focus design prototype

c = combining design prototype

However, the implications of each of these is different. Adding functions does not necessarily imply new behaviours since the existing behaviours in the focus design prototype may be sufficient to characterize the combining function. Similarly, adding behaviours does not necessarily imply either new functions or new structures. The functions in the focus design prototype may be sufficient to include the combining behaviour as a characterization. The structures in the focus design prototype may be able to produce the new behaviour.

Normally structure variables are added. If they are homogeneous variables then no change is required in behaviour. If they are heterogeneous variables then a change in behaviour may be required.

The structure is described by a set of structure variables, SV, which describe the elements of the structure and their relationships. Thus, if S_f is represented by the

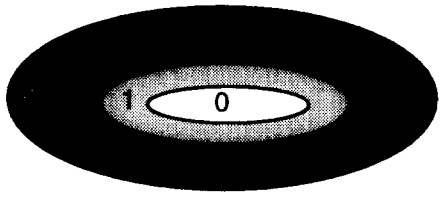

Fig. 13. Additive schemas where each successive schema entirely contains the previous schema.

set of structure variables $\{SV_{f1}, \ldots, SV_{fn}\}$ and S_c by $\{SV_{c1}, \ldots, SV_{cm}\}$, combination of S_f and S_c occurs when either some of SV_{ci} are added to SV_{fj} or in the homogeneous case SV_{ci} substitute for some SV_{fj}.

If there is only homogeneous structure variable substitution then this matches the notion of crossover in genetic algorithms [31].

Analogy is defined as the product of processes in which specific coherent aspects of the conceptual structure of one problem or domain are matched with and transferred to another problem or domain. Based on the nature of the knowledge transferred to the new problem or domain, analogical reasoning processes can be placed into one of two classes: transformational analogy and derivational analogy [32, 33].

Transformational analogy can operate on function, behaviour or structure, whilst derivational analogy operates on knowledge. The interest here is primarily on transformational analogy. Most computational analogies are drawn between situations in the same domain although interesting analogies can be drawn between situations in different domains [34]. Analogy introduces new variables into the existing design prototype called the focus design prototype from another design prototype called the source design prototype. Analogy can also be considered as a substitutive process.

Analogy can be represented as:

$$F_{new} = F_f \wedge \tau_a(F_s)$$

$$B_{new} = B_f \wedge \tau_a(B_s)$$

$$S_{new} = S_f \wedge \tau_a(S_s)$$

where

τ_a = an analogical process

s = subscript denoting source design prototype

and the other subscripts have their earlier meanings.

The implications of the results of analogical processes on function, behaviour and structure are the same as for combination alone.

Mutation is the alteration of a variable in a design prototype by an external process. Mutation can occur at two representational levels. The first is at the overt structure level [35]. As an example of the mutation of the value of structure variables, consider a structure, S, described by a set of structure variables, SV_i. Let us say that the value of SV_j which represented the

Fig. 14. Substitutive schemas where only a part (or none in the extreme) of the previous schema is contained within the current schema.

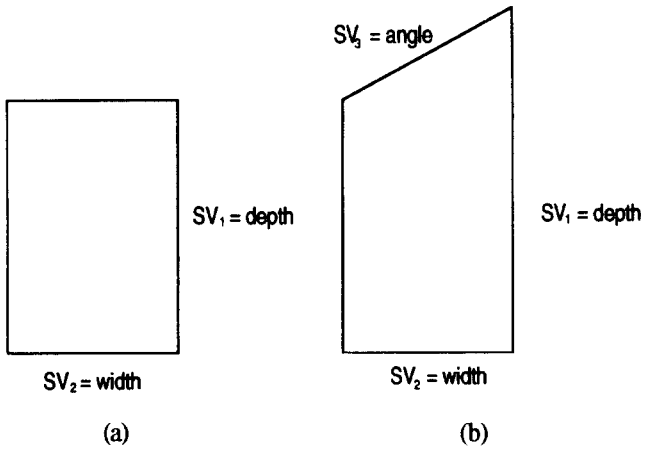

Fig. 15. (a) Original beam cross-section defined by two structure variables (SV_1, SV_2). (b) Mutated beam cross-section defined by three structure variables (SV_1, SV_2, SV_3).

connection between a door and door frame was 'hinge'. A mutation operator might change its value to 'slider'. As an example of the mutation of the structure variables rather than their values, consider the cross-section of a structural engineering beam. This beam is defined by its depth and width (SV_1, SV_2) as in Fig. 15(a). The implicit assumption is that it is rectangular. A mutation operator might change the structure variables to (SV_1, SV_2, V_3) where SV_3 is an angle as in Fig. 15(b).

At this level mutation can be represented as:

$$S_{new} = \varphi_m(S_f)$$

where

φ_m = a mutation process

Mutation draws on an analogy with genetics where the structure in a design prototype maps onto the genetic concept of phenotype whilst there is a more fundamental coding representation at the gene level called the genotype. The genotype is expressed as the phenotype through process operations. The genotype represents structure at a covert level and mutations can also occur here. At this covert level it is possible to conceive of not only structure being represented but also behaviour and function with mutation operations occurring on all three of function, behaviour and structure. At this second representational level mutation can be represented as:

$$F_{Gnew} = \varphi_m(F_{Gf})$$

$$B_{Gnew} = \varphi_m(B_{Gf})$$

$$S_{Gnew} = \varphi_m(S_{Gf})$$

where

φ_m = a mutation process

G = subscript denoting represented at gene level

and the other subscripts have their earlier meanings.

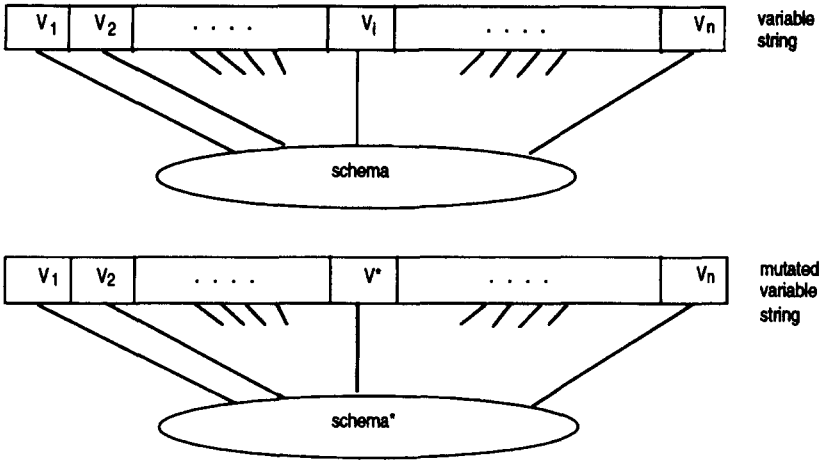

Fig. 16. Substitutive variable, V*, requires a modified schema, schema*.

Representations at the gene level form the basis of evolutionary systems to be discussed later. The simplest genotypic representation is as binary strings for possible values of variables [31]. More interestingly generative design rules can be represented in a similar manner, although the representation is not limited to binary strings. State transformation grammars have proven to be a useful gene [36]. The important issue here is that the genotypic representation allows for the mutation of the knowledge that produces structure rather than mutating the structure itself directly.

Mutation operators fall into two classes: homogeneous and heterogeneous. As before homogeneous operators are those that produce new variables of the same class as those being mutated. Heterogeneous operators are those that produce new variables of a different class to the variables being mutated. In classical genetic algorithms all mutations are homogeneous.

4.2. *Processes for the substitution of variables and their effects on schemas*

What processes with their computational analogs exist to substitute variables or even schemas? There appear to be a number of such processes of interest. Three will be considered here:

1. mutation
2. analogy
3. emergence

Mutation processes when they are heterogeneous are capable of producing substitutive variables and consequently deleting some existing variables. This is modelled in a state space representation in Fig. 4. Consider a representational string of variables with its associated mapping knowledge onto a schema. Substituting a variable in that string may potentially require a change in the schema. This is a bottom-up/data-driven approach to schema modification, Fig. 16.

Analogy processes which produce heterogeneous variables are producing substitutive variables. Analogy processes which utilise the design prototype representation also are capable of introducing elements of the source design prototype schema into the current schema.

Emergence is an important process in the substitution of schemas and is tied to concepts of design fixation [37]. Fig. 14 demonstrates the notion of substitutive schemas whilst emergence is a process for substituting

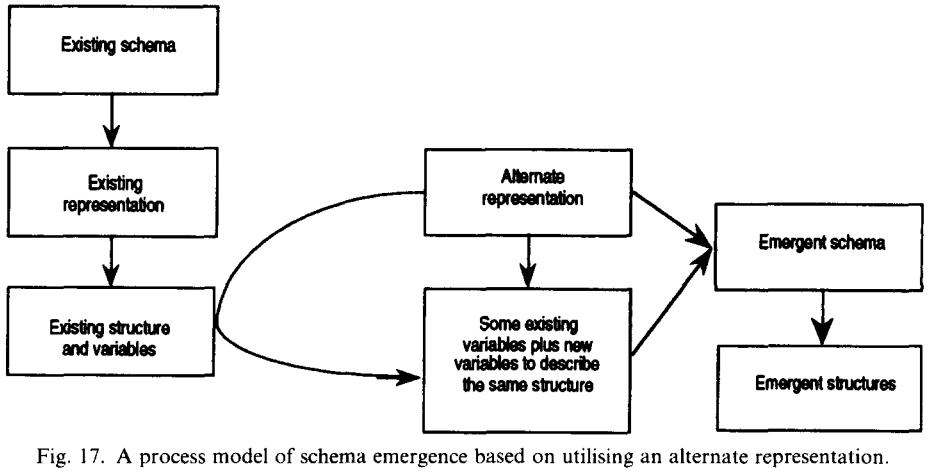

Fig. 17. A process model of schema emergence based on utilising an alternate representation.

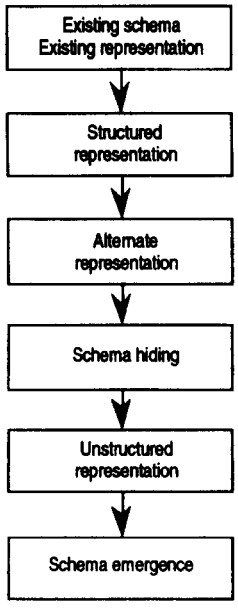

Fig. 18. Another view of a process model of schema emergence based on utilising an alternate representation.

schemas. Since emergence is the least understood and the least developed of the processes under consideration more space will be devoted to it.

How to operationalise emergence in a general domain is still in its infancy although as stated earlier there is considerable work being carried out in developing and implementing models of graphical or visual emergence. One conceptual view [38] is based on the following model. Let α be a schema, R_α be its associated representation and V_R be the variables of the schema represented in R. An alternate representation, R*, is used to represent some of the variables in V_R. This alternate representation can be associated with other schemas and as a consequence can introduce other variables. Thus, emergent schemas for apparently the same structure with its alternate representation become possible. Fig. 17 shows this in a graphical form.

Another view of schema emergence which presents the same concepts in a different manner is presented in Fig. 18.

As an example consider the three triangles in Fig. 6(a) and how they contribute to the emergent form in Fig. 6(b). Fig. 19 shows the existing schema, existing representation and existing variables.

Fig. 20 shows the new representation with its associated new variables and new schemas.

There are two ways to think about emergent schemas: searching for them and constructing them. Searching for them implies they already exist but have not been applied to the current situation. This is the top-down or hypothesis-driven approach. Use can be made of cues or features, which are sub-schemas (i.e. a small scale schema), which can be utilised to search the representation with the particular values for the variables for regularity in those features [28]. Thus, a feature in the above example might be two infinite maximal lines which intersect. This feature appears regularly as a closed shape is constructed from bounding infinite maximal lines. A shape is constructed using only this feature. However, the shape has a schema based on the characteristics of those bounding lines. Thus, an emergent schema might be four infinite maximal lines constrained such that two of them are parallel. This is sufficient to construct a schema for the trapezoid in Fig. 6(b).

5. Evolution and design

So far a model of creative design has been proposed, emergence as an important phenomenon has been discussed, and various models of creative processes presented and discussed in terms of additive and substitutive variables and additive and substitutive schemas. What all of these concepts have in common is change. The variables change, the schemas change and as a consequence novel artefacts can be designed. This notion of change leads to the concept of evolution and an analogy with natural evolution and evolutionary processes. Woodbury [39] introduced the idea of formal models of evolution in design although the concept has been discussed for some time informally [40]. The use of combination and mutation processes requires an overall design framework.

5.1. Evolutionary process model

Hybs and Gero [41] have developed a simple framework for an evolutionary process model in design. This model commences with intentions and concludes with the product or artefact and embeds design into it. It is an extension of the process model of design presented by Gero [12].

Existing schema:	**triangle shape defined by bounding line segments**
Existing representation:	**line segments**
	endpoints
Existing variables:	**x and y coordinates of endpoints**
	endpoints of bounding line segments
	bounding line segments of shapes
	shapes attached to labels

Fig. 19. Existing schema, representation and variables.

New representation:	infinite maximal lines (defined initially using the line segments above and then the line segments are discarded)
New variables:	intersections of infinite maximal lines
New schemas:	shapes defined by the number of bounding infinite maximal lines plus constraints

Fig. 20. New representation, variables and schemas.

The previously defined processes of combination and mutation are similar to the evolutionary processes which operate at the genotype level. The computational field of genetic algorithms [31] deals with this topic. Although genetic algorithms have been used to design structures [42], their use has been for routine design. The formulations have inhibited the development of creative designs by utilising highly constrained views of what could be represented, and of combination and mutation. The notion of the genotype as a fixed length string encoding covert structure needs to be extended to include all aspects of designing: function, behaviour, and some knowledge as well as structure.

The notion of schemas from genetic algorithms [43] needs to be extended to allow for a 'knowledge-rich' representation rather than its current 'knowledge-lean' representation. One way to allow for knowledge-rich representations is to encode not just the values of structure variables but also the structure variables themselves and the behaviours that are used to evaluate the fitnesses of the resulting structures. One method of encoding structure variables is to use grammars (as production systems) [39]. Section 6 describes a framework for such an approach.

5.3. Emergence and evolutionary processes in design

Traditional genetic algorithms work within a fixed schema which prevents emergence. It appears that emergence is an important concept in creative design as it is one basis for substitutive schemas.

Emergence allows for the introduction of new behaviours and new functions and is the equivalent of a designer refocusing his or her attention and/or reinterpreting the results of his or her actions so far. As an evolutionary process emergence has a number of important consequences. The most significant of these is that, in addition to new structures, new behaviours and new functions may emerge which is the equivalent of changing the environment of the phenotype since the behaviours and functions represent the environment in such systems and hence the fitness for the environment.

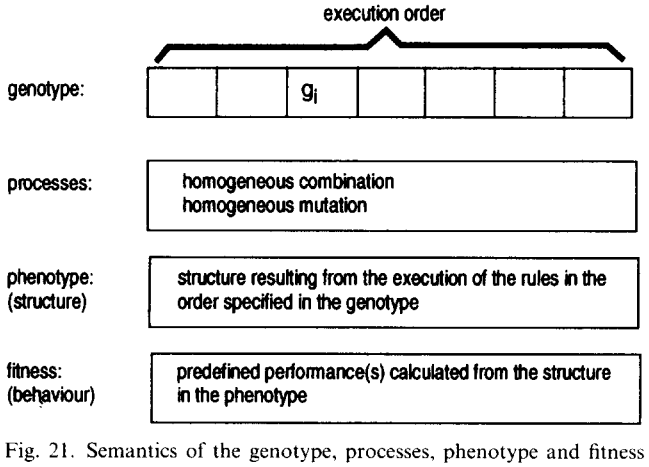

Fig. 21. Semantics of the genotype, processes, phenotype and fitness in Model-1. g_i = rule number in execution order i.

6. A computational model of evolution-based creative design

6.1. Genetic algorithm framework

Genetic algorithms provide a useful commencing framework since they already have a formal representation of various constructs and the combination and mutation operators. A number of researchers have suggested or used shape grammars as a starting point for the encoding of genes which express themselves as structure in the phenotype [36, 39, 44].

Model-1: Planning as synthesis

The first cut is to have a fixed set of shape grammar rules and encode the possible order of execution as the genes. This can be represented as in Fig. 21.

The genetic algorithm uses homogeneous combination and homogeneous mutation to determine the plan (i.e. sequence of execution of the rules) which optimizes the fitness. This is the equivalent of routine design. Planning is often an important process in design but only within the context of routine design where the fitnesses, the structures and the genes are predefined.

Model-2: Novel rules

As an extension to Model-1, the rules are also coded into the genes so that both the order of execution of the rules and the rules themselves are subject to change. This can be represented as in Fig. 22.

Here the rules can be changed by the combination and mutation processes to produce new rules which subsequently produce novel structures with improved performance(s). Of particular concern here is the legitimacy of rules which are substitutively combined (which is the way genotypes combine). This is a model of the following process.

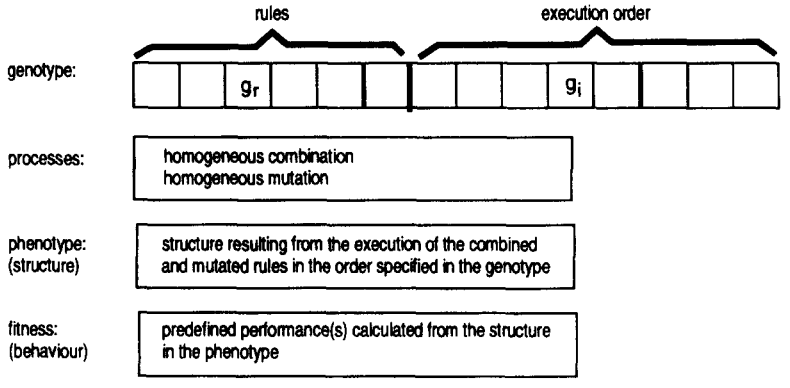

Fig. 22. Semantics of the genotype, processes, phenotype and fitness in Model-2. g_r = the rth rule; g_i = rule number in execution order i.

Given a set of rules g_r, where A is the antecedent and B is the consequent, of the form

$$A_1 \rightarrow B_1$$
$$\vdots$$
$$A_i \rightarrow B_i$$
$$\vdots$$
$$A_n \rightarrow B_n$$

Is there a way of substitutively combining them so that these new rules (the genotype) can produce new structures (the phenotype)? The new rules become

$$A_1 \rightarrow B_q$$
$$\vdots$$
$$A_i \rightarrow B_j$$
$$\vdots$$
$$A_n \rightarrow B_m$$

The effect of this is to increase the state space of the possible structures within the existing schema. Potentially, creative structures become possible. Some new rules may not be legitimate within the existing schema but may be legitimate in an alternate schema. How such an alternate schema may be either located or produced remains a research issue.

Model-3: Novel fitness

As an extension of Model-2, the fitnesses are also coded into the genes so that the order of execution of the rules, the rules themselves and the fitnesses themselves are subject to evolutionary change. This can be represented as in Fig. 23.

For each set of behaviours or fitnesses at any time in the evolutionary process the genetic algorithm will produce structures with better fitnesses generation by generation. For each modified set of behaviours or fitnesses the structures which were the improved structures for the previous set provide the commencing seed for the next generation. This becomes a non-stationary genetic algorithm.

Evolving fitnesses or behaviours is a novel task which is only now beginning to receive computational attention [45]. How new behaviours relate to the structure requires additional processes if the new behaviours contain heterogeneous variables and may possibly require additional processes if the new behaviours contain homogeneous variables only. These additional processes may be analogy to locate similar behaviours elsewhere or first principle methods to establish

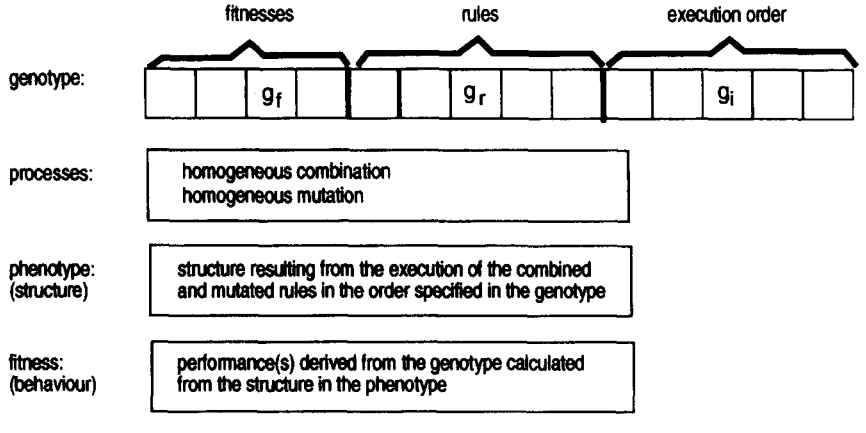

Fig. 23. Semantics of genotype, processes, phenotype and fitness in Model-3. g_f = the fth fitness; g_r = the rth rule; g_i = rule number in execution order i.

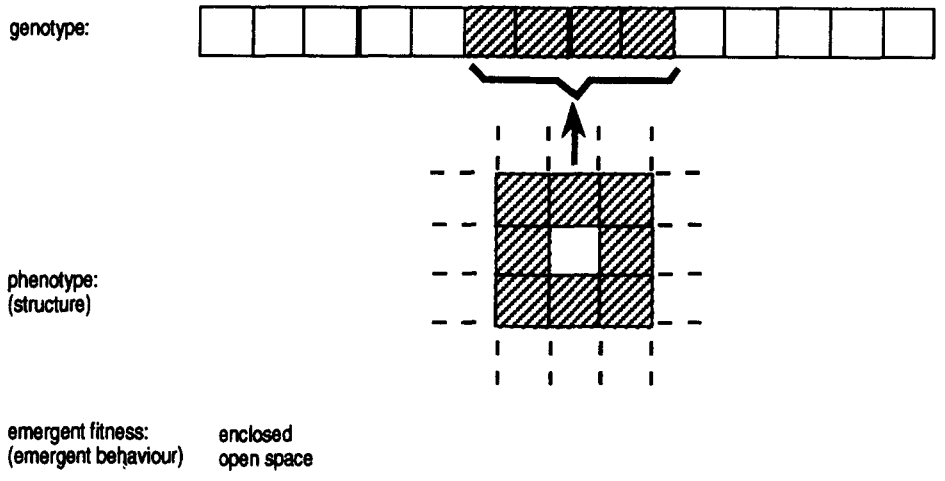

Fig. 24. Emergent behaviour of structure reverse engineered to locate and group a sequence of genes for future use.

qualitative relationships between the emergent behaviour and structure.

Model-4: Novel functions

As an extension to Model-3, the genotype could be extended to include functions as a set of genes which could be manipulated. The implications of this is hard to explore since there are very little on the representation of functions.

6.2. Emergence in an evolutionary design system

Emergence can be included in an evolutionary design system through a two-stage process:

1. emergence needs to be located; and
2. the emergent schema needs to be propagated in the system.

Locating emergence requires a process. Figs. 17 and 18 present a process model of schema emergence whilst Figs. 19 and 20 demonstrate its applicability in the visual domain. Emergent schemas can be propagated in two ways:

1. through genetic engineering; and
2. through analogy.

The genetic engineering approach [46] examines some emergent behavioural aspect of structure and reverse engineers it to locate the sequence and combination of genes which produced it. It then generates a grouping and label for that sequence so the behaviour of the identified structure can be used in a fitness test when required. Fig. 24 shows the genotype and resulting phenotype for a particular design. Using the model in Fig. 17 a new representation of the structure allows for the emergence of a schema associated with enclosed space even though the existing schema is only concerned with square elements and their boundaries. This emergent schema involves an emergent behaviour associated with enclosed open spaces. All subsequent designs are now evaluated against this additional fitness. The value of an emergent fitness is decided by the designer.

The importance of emergence in design is that it provides the opportunity to extend the state space of possible designs. As has been stated previously, emergence plays a potentially important role in design.

Analogy requires a repository of other designs suitably characterised from which analogies may be drawn. Emergent functionality is gaining importance as a research topic in artificial intelligence and a number of the approaches being tried may be applicable as starting points in the design domain [47, 48].

7. Discussion

Creative design involves the introduction of new variables which perturbate an existing schema or the emergence of new schemas. Computational processes for the introduction of new variables are sufficiently well developed to allow their use. Recognising and utilising emergence is much less well developed and is becoming an important research topic. Fundamental to all the processes is the need to recognise that the schema within which a particular variable is used is changed by additive and substitutive processes and any perturbed or emergent schema becomes the focus schema.

Design prototypes provide a meta-schema for design viewed as an evolutionary process since they play the role of the organism in natural genetics. A design prototype is both the structure and the carrier of the genetic material. It exhibits behaviour (fitness) within a context. It includes not only the genes and the knowledge about how to express the genes as structure but also the knowledge about how the structure and behaviour are connected and how the behaviour and function are connected.

There are clear research areas implied by the exposition in this paper:

1. representation of design-related genes;
2. how to represent behaviour and function;
3. development of a computational theory of emergence;
4. development of processes for locating emergence; and
5. how to understand the evolution of behaviour.

Model-1 treats design as an optimization problem where the evolutionary machinery of genetic algorithms is used to locate the values of the design decisions about structure which optimize the fitness. This matches the concept of routine design.

Model-2 allows for the evolution of new rules (the label 'rules' has been used generically rather than specifically meaning the production rule representation) for the production of new structures. This is the beginning of creative design. The processes of combination and mutation can be used to evolve new rules. Current work is addressed at implementing this model. Preliminary results indicate the utility of the model, with new rules being evolved which produce better fitnesses. The resulting fitnesses are better than the optimal fitnesses produced under Model-1. The new rules produce structures incapable of being produced with the original rules. As a side effect, this model 'learns' rules capable of generating designs not previously able to be generated. This form of learning, which will not be pursued further here, is different to learning as generalisation.

Model-3 extends the genetic coding to include the behaviours or fitnesses and as a consequence they may also evolve.

The analogy with natural evolution with its genetic substrate provides a useful computational framework in which the genetic algorithm is the process or engine used in a different manner than is customary. In design there is interest not only in synthesising solutions, even optimal solutions, but also in the novel and unexpected solution which as a consequence of its existence changes our expectations. This may require a change in structure, behaviour of function — the essence of creative design.

Acknowledgements

The ideas in this paper have benefited from discussions with many members of the Key Centre of Design Computing, in particular Jose Damski, Jun Jo, Han Jun, Vladimir Kazakov, Sushil Louis, Mary Lou Maher, Mike Rosenman, Thorsten Schnier, Weiyuan Wang and Min Yan. The girder/Goethe joke was reminded by Fay Sudweeks. This work is supported by a grant from the Australian Research Council.

References

[1] G. Broadbent, Design in Architecture, Wiley, London, 1973.
[2] P.G. Rowe, Design Thinking, MIT, Cambridge, 1987.
[3] R.D. Coyne, M.A. Rosenman, A.D. Radford, M. Balachandran and J.S. Gero, Knowledge-Based Design Systems, Addison-Wesley, Reading, 1990.
[4] B. Lawson, How Designers Think, Butterworths, London, 1990.
[5] T. Smithers, Design as exploration: puzzle-making and puzzle-solving, AID92 Workshop on Search-Based and Exploration-Based Models of Design Process (available from the Department of Artificial Intelligence, Edinburgh University), 1992, pp. 1–21.
[6] M. Boden, The Creative Mind, Myths and Mechanisms, Wiedenfeld and Nicholson, London, 1991.
[7] J.S. Gero and M.L. Maher, Mutation and analogy to support creativity in computer-aided design, in G.N. Schmitt (ed.), CAAD Futures '91, Vieweg, Wiesbaden, 1992, pp. 261–270.
[8] J.S. Gero and M.L. Maher, (eds.) Modeling Creativity and Knowledge-Based Creative Design, Lawrence Erlbaum, Hillsdale, NJ, 1993.
[9] S.H. Kim, Essence of Creativity, Oxford University Press, New York, 1990.
[10] R. Sternberg, (ed.) The Nature of Creativity, Cambridge University Press, Cambridge, 1988.
[11] R.W. Weisberg, Creativity: Genius and Other Myths, W.H. Freeman, New York, 1986.
[12] J.S. Gero, Design prototypes: a knowledge representation schema for design, AI Mag., 11 (1990) 26–36.
[13] J. de Kleer and J.S. Brown, Qualitative physics based on confluences, Artif. Intell., 24 (1984) 7–83.
[14] R. Ganesham, S. Finger and J. Garrett, Representing and reasoning with design intent, in J.S. Gero (ed.), Artificial Intelligence in Design '91, Butterworth-Heinemann, 1991, pp. 737–755.
[15] A.C.B. Garcia and C. Howard, Building a model for augmented documentation, in J.S. Gero (ed.), Artificial Intelligence in Design '91, Butterworth-Heinemann, 1991, pp. 723–736.
[16] M.A. Rosenman, Incorporating intent in design data exchange standards, in J.S. Gero and F. Sudweeks (eds.), Preprints IJCAI-91 Workshop on Artificial Intelligence in Design, University of Sydney, Sydney, Australia, 1991, pp. 51–56.
[17] M.A. Rosenman and J.S. Gero, Modelling multiple views of design objects in a collaborative CAD environment, Comput.-Aided Des., 28 (1996) 193–205.
[18] J. Beattie, An essay on laughter and ludicrous composition, Essays, William Creech, Edinburgh, 1776.
[19] A. Koestler, The Act of Creation, Hutchinson, London, 1964.
[20] J.M. Suls, A two-stage model for the appreciation of jokes and cartoons, in J. Goldstein and P. McGhee (eds.), Psychology of Humor, Academic Press, New York, 1972.
[21] S.S. Stevens, On the psychophysical law, Psychol. Rev., 64 (1957) 153–181.
[22] J.S. Gero and B. Kumar, Expanding design spaces through new design variables, Des. Stud., 14 (1993) 210–221.
[23] J.S. Gero, S. Serino and D. Choi, A computational model of emergent visual forms, Cog. Sci., 93 (1993) 85–87.
[24] R. Finke, Creative Imagery, Lawrence Erlbaum, Hillsdale, NJ, 1990.
[25] N.Y. Foo and B.P. Ziegler, Emergence and computation, Int. J. Gen. Syst., 10 (1985) 163–168.
[26] S. Forrest, (ed.) Emergent Computation, Elsevier, New York, 1990.
[27] K.R. Koedinger, Emergent properties and structural constraints: advantages of diagrammatic representations for reasoning and learning, in N.H. Narayanam (ed.), Workshop Notes — AAAI Spring Symposium on Reasoning with Diagrammatic Representations, Stanford University, Palo Alto, 1992, pp. 154–149.
[28] J.S. Gero and M. Yan, Shape emergence using symbolic reasoning, Environ. Plann. B: Plann. Des., 21 (1994) 191–212.

[29] J.S. Gero and J. Damski, Object emergence in 3D using a data driven approach, in J.S. Gero and F. Sudweeks (eds.) Artificial Intelligence in Design '94, Kluwer, Dordrecht, 1994, pp. 419–436.
[30] Y-T. Liu, Encoding explicit and implicit emergent subshapes based on empirical findings about human vision, in J.S. Gero and F. Sudweeks (eds.), Artificial Intelligence in Design '94, Kluwer, Dordrecht, 1994, pp. 401–418.
[31] D.E. Goldberg, Genetic Algorithms in Search, Optimization and Machine Learning, Addison-Wesley, Reading, MA, 1989.
[32] J.G. Carbonell, Learning by analogy: formulating and generalising plans from past experience, in R.S. Michalski, J.G. Carbonell and T.M. Mitchell (eds.), Machine Learning: An Artificial Intelligence Approach, Tioga, Palo Alto, CA, 1983, pp. 137–161.
[33] J.G. Carbonell, Derivational analogy: a theory of reconstructive problem solving and expertise acquisition, in R.S. Michalski, J.G. Carbonell and T.M. Mitchell (eds.), Machine Learning II: An Artificial Intelligence Approach, Morgan Kaufmann, Los Altos, CA, 1986, pp. 371–392.
[34] L. Qian and J.S. Gero, A design support system using analogy, in J.S. Gero (ed.), Artificial Intelligence in Design '92, Kluwer, Dordrecht, 1992, pp. 795–816.
[35] J.H. Jo and J.S. Gero, Design mutation as a computational process, in G. Woodbury (ed.), The Technology of Design, ANZAScA, University of Adelaide, Adelaide, 1991, pp. 135–143.
[36] J.S. Gero, S. Louis and S. Kundu, Evolutionary learning of novel grammars for design improvement, AIEDAM, 8 (1994) 83–94.
[37] A.T. Purcell and J.S. Gero, The effects of examples on the results of a design activity, in J.S. Gero (ed.), Artificial Intelligence in Design '91, Butterworth-Heinemann, Oxford, 1991, pp. 525–542.
[38] J. Damski and J.S. Gero, Visual reasoning as visual re-interpretation through re-representation, AID '94 Workshop on Reasoning with Shapes in Design, Lausanne, 1994, pp. 16–20.
[39] R.F. Woodbury, Design genes, Preprints Modeling Creativity and Knowledge-Based Creative Design, Design Computing Unit, Department of Architectural and Design Science, University of Sydney, Sydney, 1989, pp. 133–154. (appeared as A genetic approach to creative design in J.S. Gero and M.L. Maher (eds.) Modeling Creativity and Knowledge-Based Creative Design, Lawrence Erlbaum, Hillsdale, NJ, pp. 211–232.).
[40] P. Steadman, The Evolution of Designs, Cambridge University Press, Cambridge, 1979.
[41] I. Hybs and J.S. Gero, An evolutionary process model of design. Des. Stud., 13 (1992) 273–290.
[42] S. Louis and G.J. Rawlings, Designer genetic algorithms: genetic algorithms in structure design, in R.K. Belew and L.B. Booker (eds.), Proc. Fourth Int. Conf. on Genetic Algorithms, Morgan Kaufmann, San Mateo, 1991, pp. 53–60.
[43] J. Holland, Adaptation in Natural and Artificial Systems, University of Michigan Press, Ann Arbor, 1975.
[44] J.S. Gero and V. Kazakov, An exploration-based evolutionary model of a generative design process, Microcomput. Civ. Eng., 11 (1996) 209–216.
[45] I. Harvey, The artificial evolution of behaviour, in J.-A. Meyer and S.W. Wilson (eds.), From Animals to Animats, MIT Press, Cambridge, 1991, pp. 400–408.
[46] J.S. Gero and V. Kazakov, Evolving building blocks for design using genetic engineering: a formal approach, in J.S. Gero (ed.), Advances in formal design methods for CAD, Chapman and Hall, London, 1996, pp. 31–50.
[47] G.L. Langton, Artificial Life, Addison-Wesley, Reading, 1989.
[48] L. Steels, Towards a theory of emergent functionality, in J.-A. Meyer and S.W. Wilson (eds.), From Animals to Animats, MIT Press, Cambridge, 1991, pp. 451–461.