A framework for situated design optimization

John S Gero and Udo Kannengiesser
Key Centre of Design Computing and Cognition
University of Sydney

Key words: Design optimization, situatedness

Abstract: This paper presents a framework for situated design optimization that expands the traditional view of design optimization. It is based on the notion of interaction providing the potential for modifications of various aspects of the optimization process: problem formulation, the optimization tool, the designer and ultimately the result. In contrast to other approaches, these modifications can drive further interactions within the same optimization process. We use parts of the situated function-behaviour-structure (FBS) framework as an ontological basis to describe the effects of intertwined interactions and modifications on the state space of ongoing optimization processes.

1. INTRODUCTION

Optimization in designing is a process that aims to find the best design solution with respect to a selected set of performance criteria (Papalambros and Wilde, 2000). Optimization models are often represented in terms of a set of formal mathematical expressions:

$$\text{Given design variables } x \in \chi \subseteq \Re^n \quad (1)$$

$$\text{and constraints } h(x) = 0 \quad (2)$$

$$g(x) \leq 0 \quad (3)$$

$$\text{optimize objective } f(x) \quad (4)$$

Expression (1) defines the set of design variables as a subset of the n-dimensional real space $\Re^n$. The functions h(x) and g(x) specified by equations (2) and (3) represent equality and inequality constraints, respectively. They correspond to a set of specific performances that determine the feasibility or acceptability of the design. Finally, expression

(4) specifies an objective function f(x) to be optimized. This function defines the set of performances that determine the optimality of the design.

Table 1 shows how these mathematical descriptions map onto a common ontological representation of design objects, namely Gero's (1990) function-behaviour-structure (FBS) ontology.

Table 1. Mappings between a mathematical and an ontological model of design optimization (S = structure; B = behaviour; B^c = behaviour specified by constraints; B^o = behaviour specified by objective function)

Mathematical model	Ontological (FBS) model	
$x \in \chi \subseteq \Re^n$	S	
$h(x) = 0$	S, B^c	
$g(x) \leq 0$		
f(x)	B^o	$B = B^c \cup B^o$

Structure (S) consists of the components of a design object as well as the relationships among these components. In most design domains, structure (S) comprises the geometrical, topological and material properties of the object. Table 1 shows that the structure state space (S) of the optimum design can be mapped on the given set of design variables. Some of the constraints specify ranges of values for these variables.

Behaviour (B) is derivable from structure (S) and specifies the performance of a design object. In the optimization model, behaviour (B) is embodied in the objective function and in some of the constraint functions. Accordingly, the behaviour state space (B) of the optimum design can be viewed as the union of the set of behaviour variables defined by the objective function (Bo) and the set of behaviour variables defined by some of the constraints (Bc).

Function (F), as the teleology of a design object, is not included in design optimization models.

The FBS ontology has been used as the basis for modelling design processes as a set of transitions between function, behaviour and structure (Gero, 1990). This model is known as the function-behaviour-structure (FBS) framework. Design optimization is a specific class of design process that can be subsumed in the FBS framework, Figure 1. The FBS framework distinguishes between expected behaviour (Be) and behaviour derived from structure (Bs). Be represents a set of performance criteria used as benchmarks for the design structure (S). Bs is the set of performances that are measured or derived from structure (S). Values for Bs must be within the ranges set by Be.

While the FBS framework encompasses eight fundamental processes, the scope of optimization generally comprises only three of them, namely synthesis (labelled 2 in Figure 1), analysis (labelled 3) and evaluation (labelled 4). Synthesis in optimization generates a candidate solution as an instantiation of a point within the structure (S) state space. Analysis derives the values of the relevant behaviours of that solution using the given set of objective functions and constraints. Evaluation then compares the behaviour of different candidate solutions and decides on either continuing or stopping the search for an optimum. Most design optimization problems require iterative procedures involving large numbers of synthesis-analysis-evaluation cycles.

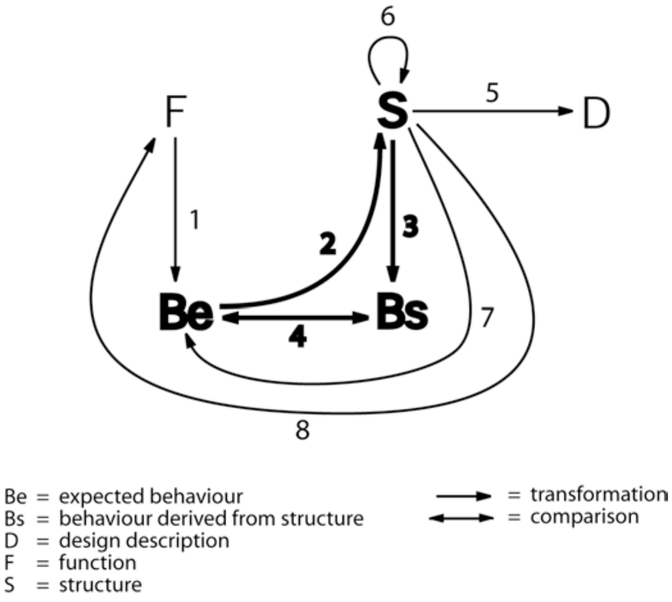

Figure 1. The processes involved in design optimization (highlighted) as a subset of the eight processes in Gero's (1990) framework of designing: (1) formulation, *(2) synthesis*, *(3) analysis*, *(4) evaluation*, (5) documentation, (6) reformulation type 1, (7) reformulation type 2, (8) reformulation type 3.

An extensive body of work exists in the development and application of optimization techniques in design across a large number of design disciplines (Wilde, 1978; Papalambros and Wilde, 2000; Pardalos and Resende, 2002; Parmee and Hajela, 2002). Most research in this area focuses on improving the speed of search for optimal designs and refining the quality of the optimal designs. This has resulted in various new optimization techniques. Recent advances are having only a marginal effect on the efficiency of most design optimization processes. This is mainly due to the following reasons:

- lack of transfer of earlier results as the design changes
- lack of domain knowledge in computational tools
- lack of task knowledge in computational tools

- lack of feedback into process strategies in the tool

These shortcomings are due to a view of design optimization that is too narrow, Figure 2. In this view, the designer starts optimization by selecting a computational tool with appropriate search methods for the particular optimization task and by formulating the problem in a tool-specific form. The tool then produces a result in terms of the structure and behaviour of an optimum design and optionally a set of post-optimality analyses. Based on that result, the designer may then reformulate the problem or tool/method selection and make repeated use of the tool to achieve a better result.

This view of design optimization does not explicitly include the grounds on which designers are able to gain experience in optimization and to decide on particular search spaces and methods. Individual experience and skills are key factors of successful design optimization but are hidden within a black box labelled the "designer". This traditional view of design optimization limits the development of better computational aids to support designers in finding optimum designs.

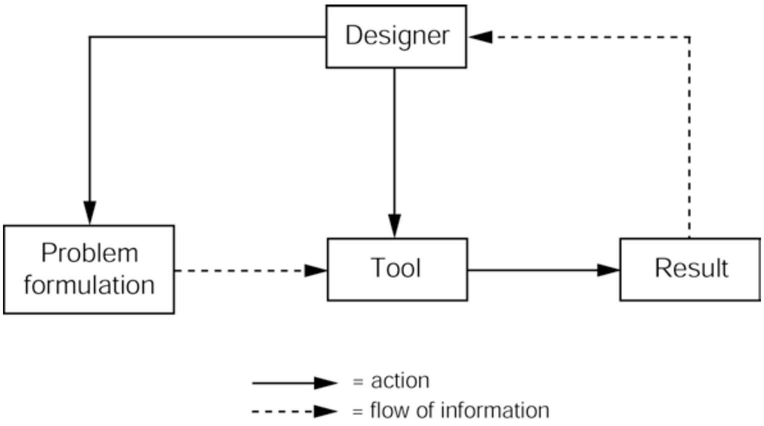

Figure 2. Traditional view of design optimization.

In this paper we propose an extended view of optimization that is based on the interaction of the computational tools, their users, the design problems and results. This provides a basis to guide further research that addresses the inadequacies described earlier. The foundations are drawn from work in situated cognition (Clancey, 1997). We then explore our situated view of design optimization using the FBS schema, which provides a basis for developing new optimization tools that can flexibly and efficiently reason about their interactions and adapt to their use.

2. SITUATEDNESS AND THE NOTION OF INTERACTION

Designing is an activity during which designers perform actions in order to change their environment. By observing and interpreting the results of their actions, they then decide on new actions to be executed on the environment. This means that the designer's concepts may change based on what they are "seeing", which itself is a function of what they have done. One may speak of an "interaction of making and seeing" (Schön and Wiggins, 1992). This interaction between the designer and the environment strongly determines the course of designing. This idea is called situatedness, whose foundational concepts go back to the work of Dewey (1896) and Bartlett (1932).

In experimental studies of designers phenomena related to the use of sketches, which support this idea, have been reported. Schön and Wiggins (1992) found that designers use their sketches not only as an external memory, but also as a means to reinterpret what they have drawn, thus leading the design in a new direction. Suwa et al. (1999) noted, in studying designers, a correlation of unexpected discoveries in sketches with the invention of new issues or requirements during the design process. They concluded that "sketches serve as a physical setting in which design thoughts are constructed on the fly in a situated way".

An idea that fits into the notion of situatedness has been proposed by Dewey in 1896 (Clancey, 1997) and is today called constructive memory. Its relevance in the area of design research has been shown by Gero (1999). Constructive memory is best exemplified by a quote from Dewey via Clancey: "Sequences of acts are composed such that subsequent experiences categorize and hence give meaning to what was experienced before". The implication of this is that memory is not laid down and fixed at the time of the original sensate experience but is a function of what comes later as well. Memories can therefore be viewed as being constructed in response to a specific demand, based on the original experience as well as the situation pertaining at the time of the demand for this memory. Therefore, everything that has happened since the original experience determines the result of memory construction. Each memory, after it has been constructed, is added to the existing knowledge (and becomes part of a new situation) and is now available to be used later, when new demands require the construction of further memories. These new memories can be viewed as new interpretations of the augmented knowledge.

The advantage of constructive memory is that the same external demand for a memory can potentially produce a different result at a later time, as newly acquired experiences may take part in the construction of that memory. Constructive memory can thus be seen as the capability to integrate new experiences by using them in constructing new memories. As a result, knowledge “wires itself up” based on the specific experiences it has had, rather than being fixed, and actions based on that knowledge can be altered in the light of new experiences.

Situated designing uses first-person knowledge grounded in the designer’s interactions with the environment (Bickhard and Campbell, 1996; Clancey, 1997; Ziemke, 1999; Smith and Gero, 2005). This is in contrast to static approaches that attempt to encode all relevant design knowledge prior to its use. Evidence in support of first-person knowledge is provided by the fact that different designers are likely to produce different designs for the same set of requirements. The same designer is likely to produce different designs at different points in time even though the same requirements are presented. This is a result of the designer acquiring new knowledge while interacting with their environment.

Gero and Kannengiesser (2004) have modelled situatedness as the interaction of different worlds, including the designer’s internal and external world. The internal world contains the designer’s experience and goals, while the external world consists of representations of things outside of the designer. Each world can bring about changes in the other world. In addition, constructive memory provides the opportunity to modify the internal world without the need for a changed external environment. The foundation of this ability is established by the designer interacting with their memories.

The notion of interaction has been shown to be central in the concept of situatedness. Interaction may also play a role in non-situated views of the (design) world; here, however, this notion is generally interpreted in terms of a simple feedback loop to inform the actions of a well-defined system that remains itself unchanged. Changes are restricted to take place only in the external environment. In contrast, a situated view allows both entities engaged in an interaction to be affected by change. This creates the potential to emerge new situations for both external and internal environments that could not have been possible with the static, non-situated model.

An important aspect in situated interaction is the notion of interpretation. Rather than being conceived of as a simple flow of information, interpretation is understood as a form of action, originating from both the external and internal environment and resulting in changes in the internal environment. Gero and Fujii’s (2000) model of interpretation as intertwined *push-pull* processes can be understood in this sense. A *push process* can be seen as an action driven by the external world aiming to change the internal world according to the data provided by the external world. A *pull process* can be seen as an action driven by the internal world aiming to change itself according to biases provided by the current expectations and goals. The notion of re-interpretation is often used to denote interpretations in which a changed internal world is the major driver of self-directed ("pull") actions leading to substantial changes in the same internal world.

3. AN INTERACTION-BASED VIEW OF DESIGN OPTIMIZATION

Integrating the notion of interaction into a model of design optimization addresses the shortcomings identified in Section 1, as it provides the opportunity for change – both in the internal "knowledge" of an optimization system and in the design it is operating on. This is a condition for future optimization tools to flexibly acquire task knowledge and domain knowledge that is adapted to the classes of optimization problems they are exposed to.

Figure 3 introduces the concept of interaction as the key element of a design optimization system. It establishes the relationships between the designer, the problem formulation, the tool and the result, which were connected previously by unidirectional arrows representing flows of information and action, Figure 2. This concept broadens the scope of optimization as a system in which every component has the potential to induce changes in other components as well as to modify itself. Static flows of information and action may now be viewed as being subsumed in the more general notion of interaction.

The interaction component in Figure 3 stands for three classes of interactions:

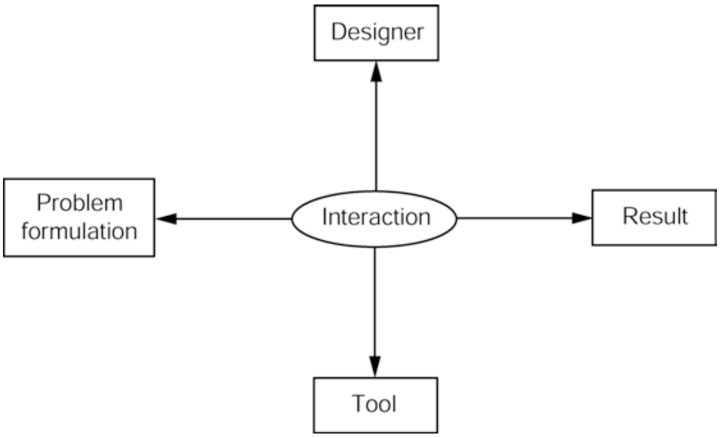

Figure 3. Interaction-based view of design optimization.

1. Interactions involving only one component: These interactions involve the designer or, alternatively, the tool as a component that interacts with itself. This includes the concept of reflective reasoning about one's own actions.
2. Interactions involving two components: Here all pairwise combinations of individual components can potentially interact. These are:
    a. interactions between the designer and the problem formulation
    b. interactions between the designer and the tool
    c. interactions between the designer and the result
    d. interactions between the problem formulation and the result
    e. interactions between the tool and the problem formulation
    f. interactions between the tool and the result
3. Interactions involving more than two components: These interactions can be viewed as involving composites rather than individual components. Numerous composites and interactions among them are conceivable. An important class of composites represents processes as they can be viewed as input-transformation-output triplets (Gero and Kannengiesser, 2006). A typical triplet consists of the problem formulation (input), the tool (transformation) and the result (output). This triplet represents a search process with which the designer or the tool may interact.

A number of previous models and systems of design optimization can be recast into such an interaction-based approach. Some of them map onto interactions between the problem formulation and the result. For example, Parmee's (1996) cluster-oriented genetic algorithms (GAs) produce preliminary results indicating high-performance regions within the search space from which further features such as sensitivity information are extracted. These features form the basis for concentrating search on particular areas within the search space via reformulating the problem in terms of structure (S) variables and constraints. Mackenzie and Gero (1987) have induced rules to detect certain features of Pareto optimal sets relating to curvature, sensitivity and other information. The rules use this information to reformulate the problem by reducing the behaviour (B) state space. As both components in the interaction between the problem formulation and the result are non-persistent, there is no place for new knowledge to be constructed and maintained. As a result, the execution of future optimization tasks cannot be directly improved without manual intervention.

Other research involves the tool as a persistent component to acquire and reuse new knowledge. A tool developed by Schwabacher et al. (1998) extracts characteristics of optimization results and uses them to formulate new optimization problems. These characteristics include information such as optimal design structure, mappings between structure and behaviour, infeasible behaviour and active constraints. This leads to better problem formulation in terms of reduced search spaces and improved starting points for gradient-based search. Jozwiak's (1987) system learns inactive constraints in order to predict whether or not the computation of particular constraint functions of a future optimization task may be neglected in the tool. Nath and Gero (2004) use machine learning to let a system acquire strategic knowledge as mappings between past design contexts and design decisions that led to useful results. These mappings are then available for the system to achieve solutions to similar design tasks more efficiently. Stahovich (2000) developed a tool that can extract iterative search strategies used by a designer. The tool then reuses these strategies to perform similar design tasks.

In most of this work, any change in the tool takes effect only after the optimization process is completed, i.e. an increase in efficiency can be achieved only in subsequent optimizations. This differentiates the cited work from the concept of situatedness, where changes have an effect on the same instance of interaction that they originate from. Situated design optimization has the potential to modify state spaces and strategies during the process of iterative search. Visualisation systems for process stages have been developed that can be recast to look like this view of design optimization (Ellman et al., 1998). The main driver for interactive change in these systems is still located inside the black box called the "designer".

While the notion of interaction as presented in Figure 3 can substantially shift our understanding of design optimization, it does not distinguish a situated view from the previous work mentioned above. A situated view includes the potential for reformulation of the structure and behaviour during the ongoing process of designing. These two types of reformulation are depicted in Figure 1 as processes 6 and 7. However, Figure 1 does not explicitly include the concept of interaction. Section 4 develops a framework for situated design optimization that is based on both notions, the one of interaction and the one of reformulation.

4. AN ONTOLOGY FOR SITUATED DESIGN OPTIMIZATION

Let us have a closer look at Gero and Kannengiesser's (2004) model of situatedness, Figure 4(a). It subdivides the internal world into an interpreted and an expected world, the latter of which is a subset of the former. These two worlds are connected to each other by a process of focussing on some of the concepts located in the interpreted world and using them as goals that are then located in the expected world. The goals are subsequently used to inform actions changing the external world.

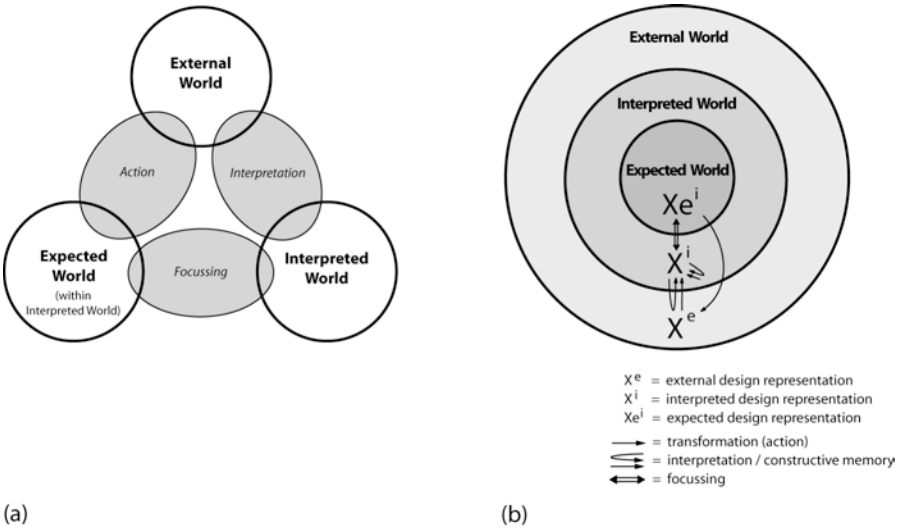

Figure 4. Situatedness as the interaction of three different worlds: (a) general model, (b) specialised model for design representations (after Gero and Kannengiesser (2004)).

Figure 4(b) presents a specialised form of this view implying a designer or design agent (as the internal world) located within the external world and placing general classes of design representations into the resultant "onion" model. The set of expected design representations (Xe^i) corresponds to the notion of a design state space, i.e. the state space of all possible designs that satisfy the set of requirements. This state space can be modified during the process of designing by transferring new interpreted design representations (X^i) into the expected world and/or transferring some of the expected design representations (Xe^i) out of the expected world. This leads to changes in external design representations (X^e), which may then be used as a basis for re-interpretation changing the interpreted world. Novel interpreted design representations (X^i) may also be the result of constructive memory, which can be viewed as a process of interaction among design representations within the interpreted world rather than across the interpreted and the external world. Both interpretation and constructive memory are viewed as push-pull processes.

The explicit integration of an expected world into a model of interaction accounts for situated designing, as changes in the internal and external world provide the grounds for further changes of the current design process via reformulations of the design state space. It can now be distinguished from other approaches that can be viewed as interaction-based but inconsistent with the idea of reformulating current goals or expectations.

Gero and Kannengiesser (2004) have used this model of situatedness as a basis for a process framework of situated designing derived from Gero's (1990) original FBS framework. Figure 5 presents those parts of their process framework that are relevant in situated design optimization. The three fundamental processes in design optimization, namely synthesis, analysis and evaluation, Figure 1, can now be viewed as consisting of partial processes described as follows:

- *synthesis*: consists of the transformation of expected behaviour (Be^i) into expected structure (Se^i) (process 1 in Figure 5) and the subsequent transformation of that structure into an external structure (S^e) by means of action (process 2).
- *analysis*: consists of the interpretation of the external structure (S^e) to produce an interpreted structure (S^i) (process 3) and the subsequent transformation of that structure into an interpreted behaviour (B^i) (process 4).
- *evaluation*: consists of the comparison of interpreted behaviours (B^i) and expected behaviours (Be^i) (process 5).

These three fundamental processes are no longer viewed as static, as they operate on design representations that are constructed on the basis of situated interaction. As a consequence, there is potential for changes in what is synthesised (i.e., what structure) and in what the design is analysed and evaluated for (i.e., what behaviour). This potential is most significant for optimization tasks that require iterative search procedures involving extensive interaction between expected, interpreted and external structures and behaviours. Situated optimization can be viewed as a process that can modify what it searches as well as what it searches for as it proceeds through the design state space. The design state space relevant for optimization is the union of the structure state space and the behaviour state space, which are represented in Figure 5 in terms of expected structure (Se^i) and expected behaviour (Be^i). Both can be modified at any time during optimization, which is represented by processes 6 and 5 in Figure 5.

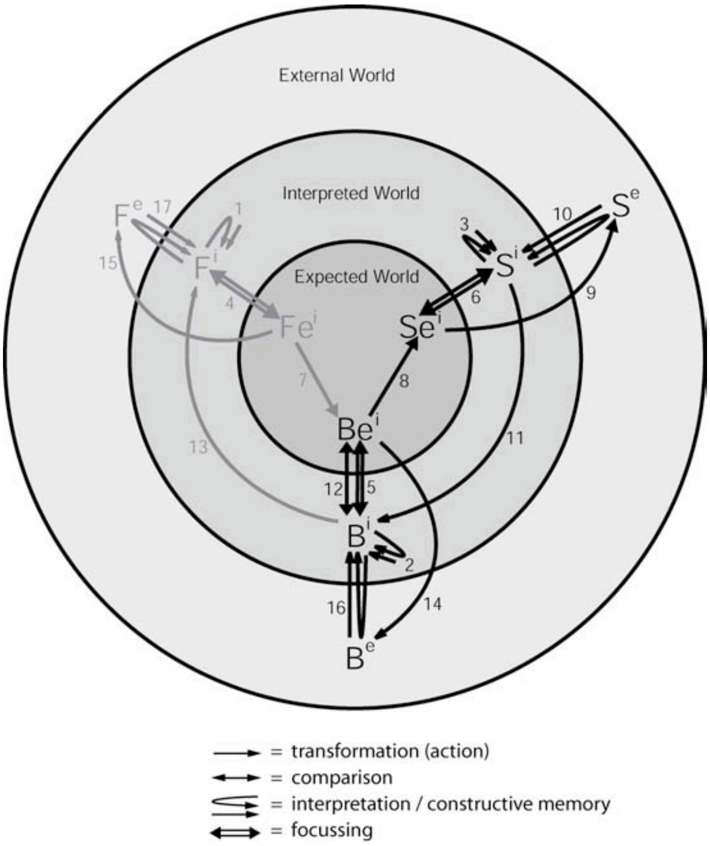

Figure 5. An ontological framework for situated design optimization

Our framework covers opportunities for change in all relevant aspects of situated optimization: change in the current design state space (expected world), change in the designer's experience (interpreted world) and change in the external design representation (external world). In the following, we present a set of instances for each of these changes (denoted by the symbol Δ) relating to the structure and behaviour of the problem formulation and the result. Examples for changes relating to other components or combinations of components proposed in Figure 3 will be included in a forthcoming paper.

Changes in the interpreted world:

- ΔS^i (represented by processes 10 and 3 in Figure 5):
  - ΔS^i of the result:
    The interpreted structure of the result can be generated externally (process 10) or internally (process 3). An example for new S^i is the substitution of a set of design variables by another variable, such as replacing 'length' and 'width' by 'aspect ratio'. These

    changes are often driven by visual design representations displayed by the tool.
  - ΔS^i of the problem formulation:
    Substitutions of existing structure variables, such as described for the result of an optimization process, may be used to simplify the problem formulation. Additional variables and their ranges of values are often constructed to complement the explicit specifications given to the designer. These variables represent implicit requirements and depend on the designer's subjective experience gathered through previous optimization tasks.
- ΔB^i (represented by processes 16 and 2):
  - ΔB^i of the result:
    New behaviours may be constructed from the results of simulations carried out by the tool that "remind" the designer of additional performances relevant for the particular optimization task. Behaviours may also be constructed internally from design cases retrieved from the designer's memory. In particular, they carry information about the location of optima of similar, past designs.
  - ΔB^i of the problem formulation:
    Initial problem formulations are often represented using natural language expressions, which may give rise to different interpretations regarding the formal set of behaviours required. In addition, a large set of implicit behaviours is often required besides those that are explicitly represented. Implicit behaviours are constructed from existing experience (i.e. the designer's implicit domain knowledge) or from interactions occurring during the current optimization process. Examples are similar to those described for ΔB^i of the result.

Changes in the expected world:
- ΔSe^i (represented by process 6):
  - ΔSe^i of the result:
    This includes the construction of predicted ranges of values for the location of the optimum design solution within the structure state space. They create a subspace representing the designer's implicit assumptions about the result of a routine optimization task that belongs to a well-defined class of similar tasks. These assumptions will be adapted over time to the designer's growing experience and the interpretation of the current optimization process.
  - ΔSe^i of the problem formulation:

This involves focussing on a set of structure variables and their ranges of values. They represent the state space of all feasible optimum design solutions. Reformulations of the structure state space occur as a result of the designer's increasing domain and task knowledge and the current state of the optimization process.

- ΔBe^i (represented by process 5):
    - ΔBe^i of the result:
      Here the designer assumes an area inside the behaviour state space, which sets predicted bounds for the performance values of the optimum design solution. It forms a subspace that is sensitive to the designer's interaction with previous and current optimization tasks. Predicted behavioural values are used to evaluate the adequacy of the results produced by the tool and of the search techniques employed.
    - ΔBe^i of the problem formulation:
      This involves focussing on a set of behaviour variables and their ranges of values according to all explicit and implicit constraints. Reformulations of the behaviour state space occur as a result of the designer's increasing domain and task knowledge and the current state of the optimization process.

Changes in the external world:

- ΔS^e and ΔB^e (represented by processes 9 and 14):
    - ΔS^e and ΔB^e of the result:
      Although the external representation of the structure and behaviour of optimization results is ultimately produced by the tool, it is part of an action that is informed by the designer's individually constructed expectations and strategies. Results form a basis for new expectations and strategies to be constructed and used to commence new cycles of synthesis, analysis and evaluation.
    - ΔS^e and ΔB^e of the problem formulation:
      This is the direct effect of an action devised by the designer to externally specify a state space of possible designs among which the tool must select the best performing one. Reformulation of the problem can be a consequence of the results of this selection process.

5. DISCUSSION

We have proposed a framework of situated design optimization that represents a departure from previous models as it is based on interaction rather than only on static relationships between the individual components of optimization. Our framework is also distinguished from other approaches by its ability to capture changes in how the optimization process proceeds as a result of its trajectory through the design state space. Using an ontological approach provides a basis for a better understanding and common ground about a situated view of optimization.

The FBS ontology has been shown to be a foundation for representing design knowledge in a general and uniform way (Gero, 1990). This is beneficial for implementing situated design optimization systems that require learning and adaptation to novel design situations. Both knowledge about the object to be optimized and knowledge about optimization strategies can be represented using the FBS schema. Current work in our research centre focuses on the development, implementation and testing of a system using this approach.

ACKNOWLEDGEMENTS

This research is supported by a grant from the Australian Research Council, grant no. DP0559885 – Situated Design Computing.

REFERENCES

Bartlett, F.C., 1932 reprinted in 1977, *Remembering: A Study in Experimental and Social Psychology*, Cambridge University Press, Cambridge.

Bickhard, M.H. and R.L. Campbell, 1996, "Topologies of learning", *New Ideas in Psychology*, 14(2), p. 111-156.

Clancey, W.J., 1997, *Situated Cognition: On Human Knowledge and Computer Representations*, Cambridge University Press, Cambridge.

Dewey, J., 1896 reprinted in 1981, "The reflex arc concept in psychology", *Psychological Review*, 3, p. 357-370.

Ellman, T., J. Keane, A. Banerjee and G. Armhold, 1998, "A transformation system for interactive reformulation of design optimization strategies", *Research in Engineering Design*, 10(1), p. 30-61.

Gero, J.S., 1990, "Design prototypes: A knowledge representation schema for design", *AI Magazine*, 11(4), p. 26-36.

Gero, J.S., 1998, "Conceptual designing as a sequence of situated acts", in: Smith (ed.) *Artificial Intelligence in Structural Engineering*, Springer-Verlag, Berlin, p. 165-177.

Gero, J.S., 1999, "Constructive memory in design thinking", in: Goldschmidt and Porter (eds.) *Design Thinking Research Symposium: Design Representation,* MIT, Cambridge, MA, p. 29-35.

Gero, J.S. and H. Fujii, 2000, "A computational framework for concept formation for a situated design agent", *Knowledge-Based Systems*, 13(6), p. 361-368.

Gero, J.S. and U. Kannengiesser, 2004, "The situated function-behaviour-structure framework", *Design Studies*, 25(4), p. 373-391.

Gero, J.S. and U. Kannengiesser, 2006, "A function-behaviour-structure ontology of processes", *Design Computing and Cognition'06*, Springer-Verlag, Berlin, to appear

Jozwiak, S.F., 1987, "Improving structural optimization programs using artificial intelligence concepts", *Engineering Optimization*, 12, p. 155-162.

Mackenzie, C.A. and J.S. Gero, 1987, "Learning design rules from decisions and performances", *Artificial Intelligence in Engineering*, 2(1), p. 2-10.

Nath, G. and J.S. Gero, 2004, "Learning while designing", *Artificial Intelligence for Engineering Design, Analysis and Manufacturing*, 18(4), p. 315-341.

Papalambros, P. and D.J. Wilde, 2000, *Principles of Optimal Design: Modeling and Computation*, Cambridge University Press, Cambridge.

Pardalos, P.M. and M.G.C. Resende (eds), 2002, *Handbook of Applied Optimization*, Oxford University Press, New York.

Parmee, I.C., 1996, "Towards an optimal engineering design process using appropriate adaptive search strategies", *Journal of Engineering Design*, 7(4), p. 341-362.

Parmee, I.C. and P. Hajela (eds), 2002, *Optimization in Industry*, Springer-Verlag, London.

Schön, D.A., 1983 *The Reflective Practitioner*, Harper Collins, New York.

Schön, D.A. and G. Wiggins, 1992, "Kinds of seeing and their functions in designing", *Design Studies*, 13(2), p. 135-156.

Schwabacher, M., T. Ellman and H. Hirsh, 1998, "Learning to set up numerical optimizations of engineering designs", *Artificial Intelligence for Engineering Design, Analysis and Manufacturing*, 12, p. 173-192.

Smith. G.J. and J.S. Gero, 2005, "What does an artificial design agent mean by being 'situated'?", *Design Studies*, 26(5), p. 535-561.

Stahovich, T.F., 2000, "LearnIT: An instance-based approach to learning and reusing design strategies", *Journal of Mechanical Design*, 122(3), p. 249-256.

Suwa, M., J.S. Gero and T. Purcell, 1999, "Unexpected discoveries and s-inventions of design requirements: A key to creative designs", in: Gero and Maher (eds.) *Computational Models of Creative Design IV,* Key Centre of Design Computing and Cognition, University of Sydney, Sydney, Australia, p. 297-320.

Wilde, D.J., 1978, *Globally Optimal Design*, Wiley, New York.

Ziemke, T., 1999, "Rethinking grounding", in: Riegler, Peschl and von Stein (eds.) *Understanding Representation in the Cognitive Sciences: Does Representation Need Reality?*, Plenum Press, New York, p. 177-190.