

AN ONTOLOGY OF COMPUTER-AIDED DESIGN

UDO KANNENGIESSER

NICTA, Australia

and

JOHN S GERO

Krasnow Institute for Advanced Study and Volgenau School of Information Technology and Engineering, George Mason University, USA, and University of Technology, Sydney, Australia

Abstract. This chapter develops an ontology of computer-aided design, based on the function-behaviour-structure (FBS) ontology. It proposes two complementary views of the process of design. The object-centred view applies the FBS ontology to the artefact being designed. Integrating an ontology of three “design worlds”, this view establishes a framework of designing as a set of transformations between the function, behaviour and structure of the design object, driven by interactions between the three design worlds. Building on this framework, the process-centred view applies the FBS ontology to the activities defined by the object-centred view. This increases the level of detail and provides a more well-defined set of representations of these activities. Our ontological framework can be used to provide a better understanding of the functionalities required of existing and future computer-aided design support.

1. Introduction

The notion of computer-aided design can be understood as an umbrella term for approaches to using computational tools to support human design activities. Its principal innovations to date include tools for computer-aided drafting (CAD), engineering (CAE) and manufacturing (CAM), which have been recognised as a significant technological achievement of the past century (Weisberg 2000). These tools are now indispensable for practitioners in many design domains.

Some research has focused on expanding computer support to activities carried out in the early, conceptual stages of design. However, its impact on design practices and tool development in industry has generally been rather

low. We believe that one of the reasons is that many of these approaches are based on an insufficient understanding of design. This concerns activities that are carried out by human designers, which include producing and re-interpreting drawings or sketches, and reflecting on current and previous design tasks. It has been shown that these activities are important drivers of designing (Schön and Wiggins 1992; Suwa et al. 1999; Suwa and Tversky 2002). Most traditional models of design are inadequate because they do not explicitly account for these findings.

Progress on a more comprehensive understanding of designing has been made only recently. Our situated function-behaviour-structure (FBS) framework (Gero and Kannengiesser 2004) represents designing as a situated act that is driven by the interactions between the designer and their environment. It uses a perspective that is oriented to the object being designed, so that designing can be shown as a set of transformations between the function, behaviour and structure of the artefact. The situated FBS framework has shown its potential to enhance human understanding of designing. This chapter develops extensions to this framework that provide a more detailed ontological basis on which computer-aided design support can be built.

Section 2 presents the situated FBS framework and shows how it derives from an object-centred view of designing driven by the interactions between three “design worlds”. Section 3, adopting a process-centred view of designing, applies the FBS ontology to the design activities defined in Section 2. This adds a significant amount of detail and rigour to the representation of each activity. Section 4 uses this view to derive a framework that specifies the functions required of computational tools to support designing. Section 5 concludes the chapter.

2. An Object-Centred Ontology of Design

2.1. AN ONTOLOGY OF DESIGN OBJECTS

Most design models and design ontologies focus on the artefact or object of design. The FBS ontology distinguishes between three aspects of a design object (Gero 1990; Gero and Kannengiesser 2004): function (F), behaviour (B) and structure (S).

2.1.1. Object Function

Function (F) of an object is defined as its teleology (“what the object is for”). For example, some of the functions of a window include “to provide view”, “to provide daylight” and “to provide rain protection”. Function represents the usefulness of the object for another system.

2.1.2. *Object Behaviour*

Behaviour (B) of an object is defined as the attributes that can be derived from its structure (“what the object does”). Using the window example, behaviours include “thermal conduction”, “light transmission” and “direct solar gain”. Behaviour provides operational, measurable performance criteria for comparing different objects.

2.1.3. *Object Structure*

Structure (S) of an object is defined as its components and their relationships (“what the object consists of”). The structure of physical objects includes their form (i.e., geometry and topology) and material. More generally, form can be viewed as a description of an object’s macro-structure, and material can be viewed as a shorthand description of the micro-structure. In the window example, macro-structure (form) includes “glazing length” and “glazing height”, and micro-structure (material) includes “type of glass”.

2.1.4. *Relationships between Object Function, Behaviour and Structure*

Humans construct relationships between function, behaviour and structure through experience and through the development of causal models based on interactions with the object. Specifically, function is ascribed to behaviour by establishing a teleological connection between the human’s goals and observable or measurable effects of the object. There is no direct relationship between function and structure. Behaviour is causally related to structure, i.e. it can be derived from structure using physical laws or heuristics. This may require knowledge about external effects (exogenous variables) and their interaction with the artefact’s structure. In the window example, deriving the behaviour “light transmission” requires considering external light sources.

2.2. AN ONTOLOGY OF DESIGN WORLDS

An aspect that has been ignored in most models of design relates to the interactions of the designer and their environment. 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. The designers’ concepts may change according to 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).

Gero and Kannengiesser (2004) have modelled situatedness by specifying three interacting worlds: the external world, interpreted world and expected world, Figure 1(a).

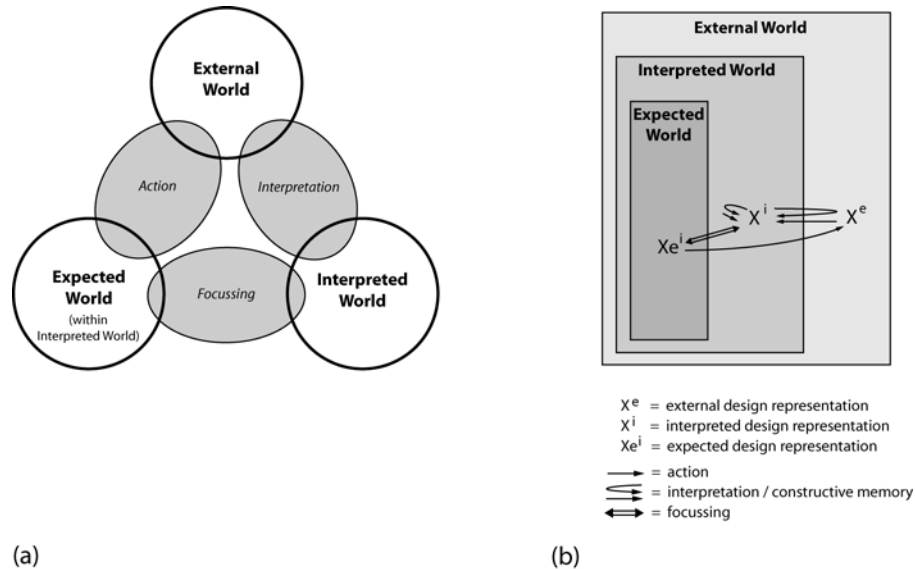


Figure 1. Situatedness as the interaction of three worlds: (a) general model, (b) specialised model for design representations

2.2.1. The External World

The external world is the world that is composed of representations outside the designer or design agent. The notion of “external” is meant in a conceptual sense rather than a physical one. It denotes an environment that contains design artefacts made available for interpretation.

2.2.2. The Interpreted World

The interpreted world is the world that is built up inside the design agent in terms of sensory experiences, percepts and concepts. It is the internal representation of that part of the external world that the design agent interacts with. The interpreted world provides an environment for analytic activities and discovery during designing.

2.2.3. The Expected World

The expected world is the world imagined actions of the design agent will produce. It is the environment in which the effects of actions are predicted according to current goals and interpretations of the state of the world.

2.2.4. Relationships between the Three Worlds

These three worlds are related through three classes of interaction. *Interpretation* transforms variables that are sensed in the external world into sensory experiences, percepts and concepts that compose the interpreted world. *Focussing* takes some aspects of the interpreted world and uses them as goals for the expected world. *Action* is an effect which brings about a change in the external world according to the goals in the expected world.

2.2.5. A More Detailed Framework of Design Interactions

Figure 1(b) presents a specialised view of the ontology of design worlds, with the design agent (described by the interpreted and expected world) located within the external world, and with general classes of design representations placed into this nested model. The set of expected design representations (X_e^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 (X_e^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 memory (here called *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, i.e. the results of these processes are driven both by the original experience (“push”) and by some of the agent’s current interpretations and expectations (“pull”) (Gero and Fujii 2000). This notion captures two ideas. First, interpretation and constructive memory have a subjective nature, using first-person knowledge grounded in the designer’s interactions with their 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. Anecdotal evidence in support of first-person knowledge is provided by the common observation that different designers perceive the same set of requirements differently (and thus produce different designs). And the same designer is likely to produce different designs at later times for the same requirements. This is a result of the designer acquiring new knowledge while interacting with their environment between the two times.

Second, the interplay between “push” and “pull” has the potential to produce emergent effects, leading to novel and often surprising

interpretations of the same internal or external representation. This idea extends the notion of biases that simply reproduce the agent's current expectations. Examples have been provided from experimental studies of designers interacting with their sketches of the design object. 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 surprising, 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". Guindon's (1990) protocol analyses of software engineers, designing control software for a lift, revealed that designing is characterised by frequent discoveries of new requirements interleaved with the development of new partial design solutions. As Guindon puts it, "designers try to make the most effective use of newly inferred requirements, or the sudden discovery of partial solutions, and modify their goals and plans accordingly".

2.3. THE SITUATED FUNCTION-BEHAVIOUR-STRUCTURE FRAMEWORK

Gero and Kannengiesser (2004) have combined the ontology of design artefacts (Section 2.1) with the ontology of design worlds (Section 2.2), by specialising the model of situatedness shown in Figure 1(b). In particular, the variable X , which stands for design representations in general, is replaced with the more specific representations F , B and S . This provides the basis of the situated FBS framework, Figure 2 (Gero and Kannengiesser 2004). In addition to using external, interpreted and expected F , B and S , this framework uses explicit representations of external requirements given to the designer by another agent (usually the customer). Specifically, there may be external requirements on function (FR^e), external requirements on behaviour (BR^e), and external requirements on structure (SR^e). The situated FBS framework also introduces the process of comparison between interpreted behaviour (B^i) and expected behaviour (Be^i), and a number of processes that transform interpreted structure (S^i) into interpreted behaviour (B^i), interpreted behaviour (B^i) into interpreted function (F^i), expected function (Fe^i) into expected behaviour (Be^i), and expected behaviour (Be^i) into expected structure (Se^i). Figure 2 uses the numerals 1 to 20 to label the resultant set of processes; however, it should be noted that they do not represent any order of execution.

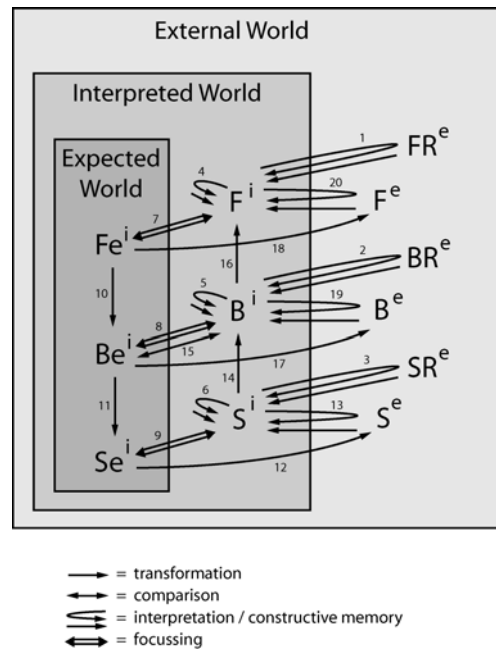


Figure 2. The situated FBS framework

The 20 processes can be mapped onto eight fundamental design steps (Gero 1990; Gero and Kannengiesser 2004).

1. *Formulation*: consists of processes 1 – 10. It includes interpretation of external requirements, given to the designer by a customer, as function, behaviour and structure, via processes 1, 2 and 3. Requirements are also constructed as implicit requirements generated from within the designer, using constructive memory (processes 4, 5 and 6). Focussing transfers a subset of the (explicitly and implicitly) required function, behaviour and structure into the expected world (processes 7, 8 and 9). In summary, processes 1 – 9 represent activities that populate the interpreted and expected worlds with design concepts, providing the basis for subsequent transformations of these concepts. Process 10 transforms expected function into additional expected behaviour. The set of expected function, behaviour and structure, resulting from the formulation step, represents the design state space. It includes all the variables and their ranges of values that are relevant for the design task.
2. *Synthesis*: consists of process 11 to generate an instance of structure that is expected to meet the required behaviour, and the externalisation of that structure via process 12. This design step can be viewed as part of a search process through the (previously formulated) state space of all possible instances of structure.

3. *Analysis*: consists of interpretation of externalised structure (process 13) and the derivation of behaviour from that structure (process 14).
4. *Evaluation*: consists of a comparison of expected behaviour and behaviour derived through analysis (process 15).
5. *Documentation*: produces an external representation of the final design solution for purposes of communicating that solution in terms of structure (process 12), and, optionally, behaviour (process 17) and function (process 18).
6. *Reformulation type 1*: consists of focussing on different structures than previously expected (process 9). Precursors of this process are the interpretation of external structure (process 13), constructive memory of structure (process 6) or the interpretation of new requirements on structure (process 3).
7. *Reformulation type 2*: consists of focussing on different behaviours than previously expected (process 8). Precursors of this process are the derivation of behaviour from structure (process 14), the interpretation of external behaviour (process 19), constructive memory of behaviour (process 5) or the interpretation of new requirements on behaviour (process 2).
8. *Reformulation type 3*: consists of focussing on different functions than previously expected (process 7). Precursors of this process are the ascription of function to behaviour (process 16), the interpretation of external function (process 20), constructive memory of function (process 4) or the interpretation of new requirements on function (process 1).

The numbering of the eight design steps, similar to the 20 processes, does not prescribe any order of execution. While it may be expected for some routine design tasks to follow a sequential execution of only the first five steps, it has been found that all three types of reformulation frequently occur throughout the process of designing (McNeill et al. 1998).

The situated FBS framework represents designing independently of the domain of the design and the specific methods used, and of the subject carrying out the process of designing. What we have referred to as the “design agent” in the definition of the three design worlds can be embodied by a human designer (or team of human designers), a computational tool, or a combination of both.

3. A Process-Centred Ontology of Design

The object-centred ontology of design presented in Section 2 has been helpful for establishing a basic understanding of design. Its emphasis on artefacts provides an intuitive, tangible perspective, representing the process of designing as a gradual evolution of the design object across three levels.

The three-world model of design interactions, in which this representation is embedded, is sufficiently rich to account for the phenomena of situatedness.

However, the object-centred ontology lacks sufficient detail and rigour to be useful for comparing or developing different methods and computer support for designers. The key ideas and semantics conveyed by the situated FBS framework are only informally expressed using textual, natural-language descriptions such as in Sections 2.2 and 2.3. The graphical model in Figure 2 does not fully capture these semantics. The mapping of the 20 processes onto Gero's (1990) eight fundamental design steps has added some more meaning by locating these processes within typical phases of a design project. However, this mapping does not completely capture all the semantics and is too informal to be used as an ontological framework for computer-aided design. What is needed is an ontology that is process-centred, treating design processes as first-class entities rather than derivatives of object-centred constructs. This Section will present such an ontology, extending our recent work on an FBS ontology of processes (Gero and Kannengiesser 2007).

3.1. AN ONTOLOGY OF PROCESSES

Processes are usually understood as entities that are less tangible than (physical) objects. Nonetheless, they can be represented using the same set of ontological constructs as used for describing objects: function, behaviour and structure. To clearly distinguish between the notations of the process-centred and the object-centred FBS ontology, we will use the indices "p" for "process" and "o" for "object".

3.1.1. *Process Function*

Function (F^p) of a process is ontologically no different to object function, as it is based on the observer's goals rather than on embodiment as an object or as a process. Instances of process functions are largely domain-dependent. However, most processes that we design and execute through actions have the general function of replacing an existing state of the world with a desired one.

3.1.2. *Process Behaviour*

Behaviour (B^p) of a process relates to attributes that allow comparison on a performance level as a basis for process evaluation. Typical process behaviours are speed, cost, amount of space required and accuracy. These behaviours can be specialised and/or quantified for instances of processes in particular domains.

3.1.3. *Process Structure*

Through an analogy with the structure of physical objects, we can distinguish between a macro- and a micro-structure (S^p) of processes.

The macro-structure of a process includes three components and two relationships, Figure 3.

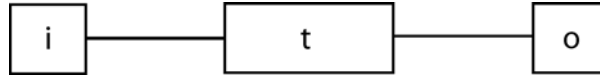


Figure 3. The macro-structure of a process (i = input; t = transformation; o = output)

The components are

- an input (i),
- a transformation (t) and
- an output (o).

The relationships connect

- the input and the transformation (i – t) and
- the transformation and the output (t – o).

Input (i) and output (o) represent properties of entities being transformed in terms of their variables and/or their values. For example, the process of transportation changes the values for the location of a (physical) object (e.g. the values of its x-, y- and z-coordinates). The process of electricity generation takes mechanical motion as input and produces electrical energy as output.

A common way to describe the transformation (t) of a process is in terms of a plan, a set of rules or other procedural descriptions. A typical example is a software procedure that is expressed in source code or as an activity diagram in the Unified Modeling Language (UML). Such descriptions are often used to specify sub-components of the transformation.

The relationships between the three components of a process are usually uni-directional from the input to the transformation and from the transformation to the output. For iterative processes the t – o relationship is bi-directional to represent the feedback loop between the output and the transformation.

The micro-structure or “material” of a process differs from the macro-structure because its components and relationships cannot be distinguished (or are not relevant) at the same level of abstraction. For example, it is not common to specify the (business process) transformation “pay the supplier” in terms of more fine-grained activities (sub-components) such as “log in to online banking system”, “fill out funds transfer form” and “click the submit button”. This set of activities is best viewed as a micro-structure specified

only through a shorthand qualifier such as “using internet banking”. Micro-structure can also be associated with the input and output components of a process. For example, a set of measuring data that is the input of a statistical analysis process may be “materialised” through either digital or paper-based media.

While micro-structure is clearly needed to carry out (“materialise”) a process, the components and relationships of that micro-structure are not explicitly represented. This fits with one of *Merriam-Webster’s* definitions of material as “the formless substratum of all things which exists only potentially and upon which form acts to produce realities”.

The “formless substratum” of a transformation may reference not only processes but also objects. It then denotes the entity or agent executing the transformation. In the “pay the supplier” example, it is possible to specify “finance officer” or “purchasing department” as a general descriptor for the executing agent. Including such references to agents (as “actors” or “roles”) has become well-established in process modelling (Curtis et al. 1992).

Since micro-structure does not specify components and relationships, it can be embodied by either (micro-) objects or (micro-) processes. In some instances, the micro-structure of objects can refer to (micro-) processes rather than (micro-) objects. For example, the chemical bonds (macro-relationships) between the atoms (macro-components) of a molecule are realised by physical processes, according to the laws of quantum electrodynamics. A view of the world as being based on processes rather than objects has generally been suggested in process philosophy (Rescher 2006).

3.1.4. Relationships between Process Function, Behaviour and Structure

Relationships among F^p , B^p and S^p are constructed according to the same principles as described for F^o , B^o and S^o (see Section 2.1.4). Function is ascribed to behaviour based on associations of process performance with human goals. Behaviour can be derived from structure either directly or indirectly based on external effects. An example of directly derived behaviour is the speed of a process, as this depends exclusively on the macro-structure (“what kind of transformation is used on what input to produce what output?”) and the micro-structure (“how and by whom is the transformation carried out using what input/output media?”). An example of indirectly derived behaviour is accuracy, which needs an external benchmark against which the output of the process is compared.

3.2. AN ONTOLOGY OF DESIGN PROCESSES

The FBS ontology of processes can be used to re-represent the object-centred description of the 20 design processes (presented in Section 2.3) as a

process-centred one. Most parts of the object-centred model depicted in Figure 2 directly map onto the input and output components of process macro-structure (S^p). For example, S^p of process 14 in Figure 2 includes (interpreted) object structure (S^o) as input (i) and (interpreted) object behaviour (B^o) as output (o).¹ No specific information is given about transformation components, as this is available only at an instance level.

Most of the semantics of the situated FBS framework can be captured by process function (F^p). Table 1 gives an overview of the structure and functions of each of the 20 design processes.

TABLE 1. Function (F^p) and macro-structure (S^p) of the 20 design processes

ID	Process class	(macro-) S^p	F^p
1	Interpretation	$FR^e \rightarrow F^i$	1. transfer design concepts as intended 2. re-interpret design concepts
2		$BR^e \rightarrow B^i$	
3		$SR^e \rightarrow S^i$	
4	Constructive memory	$F^i \rightarrow F^i$	1. retrieve design concepts as stored 2. re-construct design concepts
5		$B^i \rightarrow B^i$	
6		$S^i \rightarrow S^i$	
7	Focussing	$F^i \rightarrow Fe^i$	construct function state space
8		$B^i \rightarrow Be^i$	construct behaviour state space
9		$S^i \rightarrow Se^i$	construct structure state space
10	Transformation	$Fe^i \rightarrow Be^i$	construct behaviour state space
11		$Be^i \rightarrow Se^i$	generate values for design structure
12	Action	$Se^i \rightarrow S^e$	1. communicate the design to others 2. initiate reflective conversation
13	Interpretation	$S^e \rightarrow S^i$	1. transfer design concepts as intended 2. re-interpret design concepts
14	Transformation	$S^i \rightarrow B^i$	1. analyse for performance expectations 2. generate new design issues
15	Comparison	$\{Be^i, B^i\} \rightarrow$ decision	evaluate the design
16	Transformation	$B^i \rightarrow F^i$	generate new design issues
17	Action	$Be^i \rightarrow B^e$	1. communicate the design to others 2. initiate reflective conversation
18		$Fe^i \rightarrow F^e$	
19	Interpretation	$B^e \rightarrow B^i$	1. transfer design concepts as intended 2. re-interpret design concepts
20		$F^e \rightarrow F^i$	

Interpretation processes (1, 2, 3, 13, 19 and 20) can have two different functions. One function is to transfer existing design concepts from one

¹ Indices for “interpreted” have been omitted here to improve notational clarity.

agent to another or the same agent without a change of the initial meaning of these concepts. This involves bringing external representations into a form that allows processing of these representations by the individual design agent. The other function of interpretation is to re-interpret design concepts based on existing ones. This generates design concepts and issues that are novel with respect to the ones initially intended.

Constructive memory processes (4, 5 and 6) have a similar set of functions. One function is to retrieve design concepts from some storage space in the same way as they were experienced at the time of storage. While this may include some computation or transformation, such as refinement or decomposition of design concepts, the results of this process will all have a pre-defined relationship with the initial concepts. The other function of the constructive memory processes is to re-construct and thereby modify existing design concepts, which corresponds to the notion of reflection (Schön 1983).

Focussing processes (7, 8 and 9) have the function to construct the design state space. This includes the construction of the initial design state space (maps onto the formulation step) and subsequent modifications of that space (maps onto the reformulation steps).

Action processes (12, 17 and 18) can have two different functions. One function is to communicate aspects of the design to other stakeholders (agents). Here, the notion of communication is used in its traditional sense of sharing information, based on unambiguous transfer of design concepts. The other function is to initiate reflective conversation, either with other stakeholders (agents) or the initiator of the action process itself. In other words, external representations are produced to be re-interpreted in new ways.

Processes 10, 11, 14 and 16 may be called “FBS^o transformations” based on their role as transformers between F^o, B^o and S^o. Process 10 has the function to construct the behaviour state space, and process 11 has the function to generate values within the (previously constructed) structure state space. Process 14 has two functions. One function is to analyse the design with respect to current performance expectations. The other function is to generate new design concepts that can be included as new issues in the current design task. This is also the function of process 16. The comparison process (15) has the function to evaluate the design, based on decision making informed by comparison of expected and interpreted design performance.

It can be seen that some of the functions (F^p) – loosely speaking – relate to non-situated and others to situated aspects of designing. Non-situated aspects are captured by those functions that do not address the potential for change during designing. These are the functions that involve “transfer” (in

interpretation processes), “retrieval” (in constructive memory processes) and “communication” (in action processes). Situated aspects of designing describing the potential for change are captured by functions that involve “re-interpretation” (in interpretation processes), “re-construction” (in constructive memory processes) and “reflective conversation” (in action processes).

Table 1 does not include the behaviours (B^p) of the 20 processes. This is because, at the current level of abstraction, they are no different from the general process behaviours described in Section 3.1.2. This is based on the independence of our ontology of specific methods or design domains. No detailed information about structure (S^p) and exogenous effects is available to be able to specialise or quantify general process behaviours (B^p) such as speed, accuracy and cost. An example for such detailed information would be when process structures (S^p) were considered that contain iterations (e.g., when using genetic algorithms (GAs) in design synthesis). In this case, the behaviour (B^p) “rate of convergence” could be derived that is a specialisation of the behaviour (B^p) “speed”. However, as our aim here is to provide a general rather than an instance-specific ontology, different classes of design processes are distinguished only at the level of function (F^p) and structure (S^p).

4. An Ontological Framework for Computer-Aided Design Support

The ontological view presented in Sections 2 and 3 has provided a detailed description of 20 distinct processes in designing. This is useful for enhancing our understanding of designing as a human activity. However, the ultimate aim of most research in design is to enhance the performance of this activity, both in terms of higher effectiveness and efficiency. The key to improving performance or behaviour (B^p) of designing is in the structure (S^p) it is derived from. This requires more detailed representations of structure (S^p) than presented in Table 1, and mainly concerns micro-structure. Exploring the micro-level of process structure is a general research theme that has been recognised in a number of other disciplines (Osterweil 2005).

Research in the micro-structure (S^p) of designing can be characterised loosely as either method- or tool-oriented. Method-oriented approaches focus on process-centred representations of micro-structure. These representations can be viewed as composing a new macro-structure to be “materialised” by humans or tools. Tool-oriented design research focuses on object-centred representations of micro-structure in terms of new design tools. Computer-aided design research and development is clearly located in this field. Both method- and tool-oriented research streams are complementary, as each of them often uses results from the other.

Computer-aided design tools can themselves be regarded as design objects. Applying the FBS ontology to these tools provides a schema for the characteristics that the tools must exhibit to be useful in the process of designing. We will use the index “t” for “tool” to distinguish the FBS view of tools from the FBS view of design objects and design processes. The most essential characteristics of a tool relate to function (F^t) as they orient the specification of a tool’s behaviour (B^t) and structure (S^t) towards the required goals and context of use. Many of the functions of computer-aided design tools do not differ from any other software product. They include such general characteristics as usability, reliability, maintainability and others (ISO 2001). However, there are a number of functions that are specific to computer-aided design tools. These functions relate to the tools’ role as the “material” of design processes, and can generally be described as “to support design processes of class X”. For example, a general function (F^t) of a commercial CAD tool is “to support design processes of class X = documentation” (one of the fundamental design steps presented in Section 2.3). These functions can be further specialised using particular combinations of the FBS^p properties of the 20 design processes presented in Table 1. An example of a more specific function (F^t) of a CAD tool is “to support the process structure (S^p) $Se^i \rightarrow S^e$ in a way to achieve the process function (F^p) of communicating the design to others”.

The set of functions (F^t) derivable in this way can serve as high-level requirements for the development of new design tools. This approach makes research and development in computer-aided design look like a design process, generating computational models and architectures as the structure (S^t) of tools exhibiting certain behaviours (B^t) to achieve the required functions (F^t). The remainder of this Section will cast existing work on computer-aided design systems in this ontology, classifying that work based on tool functions (F^t) derived from combinations of F^p and S^p shown in Table 1. This aims to provide an overview of the current range of both commercial software and academic proof-of-concept demonstrators. For this purpose, detailed descriptions of their behaviour (B^t) and structure (S^t) are not required. Readers may consult our references to the literature for more specific information.

4.1. COMPUTER-AIDED DESIGN SUPPORT FOR ACTION

The notion of a “tool” has traditionally been viewed as a mechanism for humans to perform actions. Computer-aided design tools can serve two possible functions (F^t) in their support of action (see Table 1):

- to support communicating the design
- to support initiating reflective conversation

Figure 4 highlights processes 12, 17 and 18 in the situated FBS framework to represent actions related to these two functions.

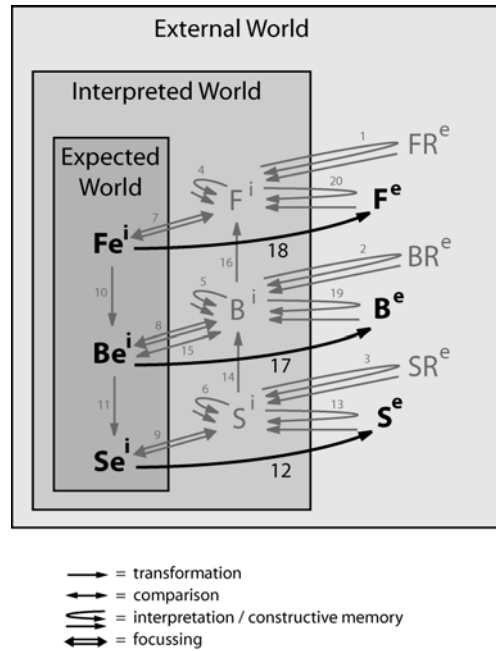


Figure 4. Action in the situated FBS framework

4.1.1. Support for Communicating the Design

- **$Se^i \rightarrow S^e$ (process 12):** The ability to generate representations of external object structure (S^e) is provided by commercial CAD systems. These tools produce 2-D or 3-D models and offer functionalities such as scaling, rotating and rendering to communicate different aspects of the object. The models generated by CAD systems are primarily used for data exchange with other designers, manufacturers or other stakeholders, or for providing input for tools that perform analyses of the designed objects. Communication across different tools has been recognised as an area of growing concern, as the tools generally use different languages (data formats) for representing object structure. A number of approaches address this problem by defining standardised product models, the best known of which are STEP and IFCs (Eastman 1999). Many CAD tools now have translators (called pre-processors) that map object structure onto a neutral format based on these standards. Some of our previous work was concerned with

developing an agent-based approach to communicating product data in situations where no standard formats are available (Kannengiesser and Gero 2006; Kannengiesser and Gero 2007).

- **$Be^i \rightarrow B^e$ (process 17):** Virtual reality (VR) systems are increasingly used to generate 3-D objects in a place-like context that usually include avatars representing potential users or stakeholders of the design. These tools support modelling not only the structure (S^o) but also the behaviour (B^o) of the designed object based on simulated interactions with avatars or other objects. Digital mock-ups (DMUs) are based on a similar concept, and are commonly used for the simulation of assembly operations or kinematics. Other tools that focus mainly on the communication of object behaviour (B^o) are those specialised in performing particular engineering analyses. Typical examples here include the representation of stresses and temperatures.
- **$Fe^i \rightarrow F^e$ (process 18):** There are currently no commercial tools specialised in generating formal representations of object function (F^o). This is mainly due to the lack of a commonly agreed representation language. In most cases, function is described informally using natural language expressions, usually based on verb-noun pairs (Jacobsen et al. 1991) that are also used in this chapter. These descriptions can be produced by general-purpose word processors and annotation mechanisms provided by CAD systems. Future tool support may result from recent work on more formal representations of function (Chandrasekaran and Josephson 2000; Stone and Wood 2000; Szykman et al. 2001; Deng 2002).

4.1.2. *Support for Initiating Reflective Conversation*

- **$Se^i \rightarrow S^e$ (process 12):** There are no commercial design tools that explicitly aim at supporting reflective conversation. However, there are some method-oriented approaches that may inform the development of such tools. For example, Jun and Gero (1997) have demonstrated how shapes can emerge by representing the same geometrical structure in different ways. Current CAD systems do not have this ability, as their representations are fixed through the way they store a design's geometry in their database. An approach by Reymen et al. (2006) uses checklists and forms for designers to stimulate the creation of textual descriptions of designs from multiple perspectives, at regular intervals during the process of design.
- **$Be^i \rightarrow B^e$ (process 17):** Reflective conversation at the behaviour level has not been well understood. However, the models of

generating multiple representations described for the structure level can be applied when behaviour is represented using shapes. The notion of space, for example, can be viewed as a behaviour (derived from a walls-and-floor structure) that can be described geometrically.

- **$Fe^i \rightarrow F^e$ (process 18):** Apart from cases in which functions represent references to shapes, reflective conversation at the function level has not been well understood. Tool support for generating multiple, textual representations of function may be developed based on research in natural language semantics. For example, de Vries et al. (2005) explore the use of the WordNet lexicon (Miller 1995) to generate a graph of synonyms and other semantic relations from a given set of words.

4.2. COMPUTER-AIDED DESIGN SUPPORT FOR FBS^o TRANSFORMATIONS AND EVALUATION

A number of research efforts have concentrated on tool support for performing those transformations and evaluations that have been viewed as fundamental in most traditional models of designing (e.g., Asimov (1962)). These include the transformations between the function, behaviour and structure of the design object, and evaluation based on comparing expected with “actual” behaviour. Figure 5 highlights processes 10, 11, 14, 15 and 16 to represent these activities.

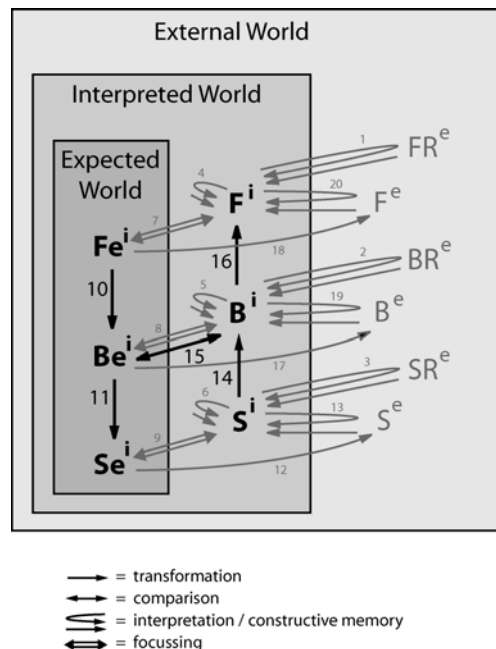


Figure 5. FBS^o transformations and evaluation in the situated FBS framework

- **$S^i \rightarrow B^i$ (process 14):** There is a wide range of commercial tools that support the derivation of object behaviour from design structure. These are commonly referred to as analysis tools or simulation tools. Most of them are based on the physical laws and principles established in the engineering sciences. Examples of design analyses for which there is automated support include finite element analysis, thermal analysis, energy analysis and kinematic analysis. Some tools, such as design optimisation tools and parametric CAD systems, provide automated support for the $S^i \rightarrow B^i$ transformation as part of a collection of transformations that also include evaluation (process 15) and the generation of object structure (process 11). These tools will be presented in more detail under the bullet points for processes 11 and 15 (below). The function of generating new design issues (see Table 1) is addressed by some CAD systems performing runtime analyses of the design, such as Design for X (DFX) analyses. Gero and Kazakov (1998) have developed a computational model of behaviour analogy where new behaviour variables are introduced into the target design based on structure similarity with the source design.
- **$Be^i \rightarrow Se^i$ (process 11):** Parametric CAD systems have shown to significantly facilitate the creation of solid models (Shah and Mäntylä 1995), and many CAD vendors now offer parametric modelling features. These systems can be viewed as automating the process of computing an object structure once a set of parameters have been formulated for both structure and behaviour. Parametric CAD systems also allow for automated maintenance of parametric constraints (Sacks et al. 2004). This requires additional automation for analysing and evaluating the design for constraint violations, which can be mapped onto the transformation process $S^i \rightarrow B^i$ (process 14) and the evaluation process $\{Be^i, B^i\} \rightarrow \text{decision}$ (process 15). Design optimisation tools provide similar integrated functionalities supporting the same set of processes. They provide an extensive range of mechanisms to evolve object structure, including various deterministic and stochastic search methods (Papalambros and Wilde 2000).
- **$\{Be^i, B^i\} \rightarrow \text{decision}$ (process 15):** Automated support for this process is provided in a number of computer-aided design systems, as indicated above. Optimisation tools, in particular, incorporate sophisticated strategies for controlling the execution of alternative search paths, based on the performance of the current design

candidate. Research on agent-based design systems addresses evaluation using conflict resolution mechanisms, which have been applied to instances of multi-objective design optimisation (Grecu and Brown 1996; Campbell et al. 1999).

- **$Fe^i \rightarrow Be^i$ (process 10):** Few systems have been developed that support the generation of object behaviours based on object function (Maiden and Sutcliffe 1992; Bhatta et al. 1994; Umeda et al. 1996). This is mainly due to the lack of a formal language to represent function.
- **$B^i \rightarrow F^i$ (process 16):** There has been no work to date on tool support for this process.

4.3. COMPUTER-AIDED DESIGN SUPPORT FOR FOCUSsing

There has been some work on tools to support focussing, the processes involved in the formulation of a design state space. These tools are mainly based on decision-making mechanisms that use various kinds of information. Figure 6 highlights processes 7, 8 and 9 to represent focussing.

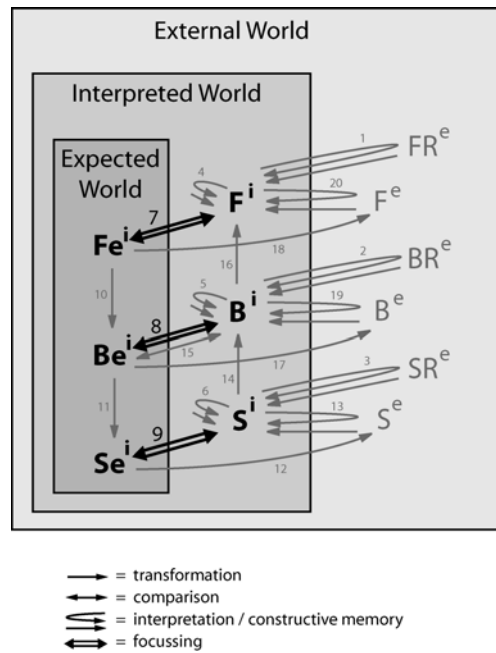


Figure 6. Focussing in the situated FBS framework

- **$S^i \rightarrow Se^i$ (process 9):** A number of computational approaches to focussing on object structure have been developed in the area of design optimisation. Some of this work uses information extracted

from the current design. For example, Parmee's (1996) cluster-oriented genetic algorithms (COGAs) identify high-performance regions within the current structure state space. These features are then used for focussing on different structure variables and constraints, to concentrate the search for an optimum design on particular areas within the original structure state space. Other work uses information learnt from previous design tasks. A tool developed by Schwabacher et al. (1998) extracts characteristics of previous optimisation results and uses them to formulate new optimization problems. These characteristics include information such as optimal structure, mappings between structure and behaviour, infeasible behaviour and active constraints. This information is used to improve the problem formulation by reducing the structure state space.

- **$B^i \rightarrow Be^i$ (process 8)**: Some work has been done on focussing at the level of object behaviour, again mostly in the context of optimisation. 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 carrying out focussing in a way that reduces the behaviour state space. Jozwiak's (1987) approach uses learning to acquire knowledge of inactive constraints, which is then used to predict whether or not the constraints of the current optimisation task may be neglected.
- **$F^i \rightarrow Fe^i$ (process 7)**: There has been no work to date on tool support for this process.

4.4. COMPUTER-AIDED DESIGN SUPPORT FOR INTERPRETATION

Tools usually have some form of interface to receive and utilise input provided externally either by humans or other tools. In computer-aided design, there are two possible functions (F^i) for interpretation by tools:

- to support transferring design concepts as intended
- to support re-interpreting design concepts

Figure 7 highlights processes 1, 2, 3, 13, 19 and 20 to represent interpretation.

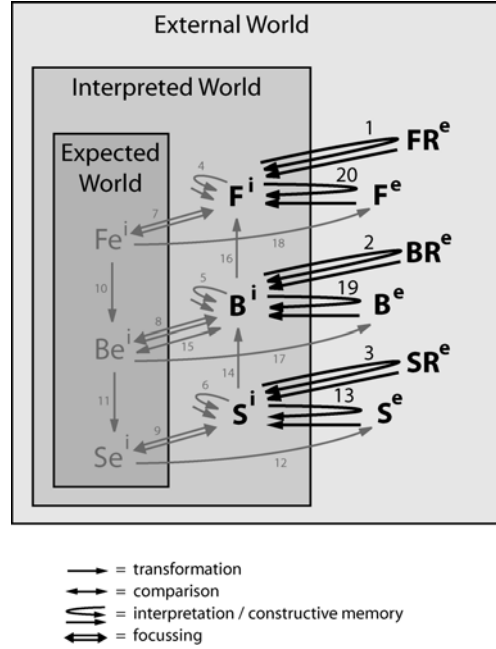


Figure 7. Interpretation in the situated FBS framework

4.4.1. Support for Transfer of Design Concepts as Intended

- **$SR^e \rightarrow S^i$ (process 3) and $S^e \rightarrow S^i$ (process 13):** There has been considerable research in the computational interpretation of external object structure. The standardisation approaches to product modelling, mentioned in Section 4.1.1, provide the basis for the development of import mechanisms (called post-processors) that translate the standard models into the tool's native format. Post-processors for STEP and IFC models are available in a number of commercial CAD/CAE/CAM systems. Another area of research is concerned with the interpretation of human sketches and freehand drawings by tools converting them into more exact graphical models or performing early design analyses (Taggart 1975; Gross 1996; Leclercq 2001).
- **$BR^e \rightarrow B^i$ (process 2) and $B^e \rightarrow B^i$ (process 19):** Most design tools dealing with object behaviour directly derive that behaviour from structure (process 14) rather than interpreting it externally (e.g., from other tools). As a result, not much work exists on tool support for the interpretation of object behaviour. However, recent approaches to interoperability aiming to standardise the representation of function and behaviour besides structure (Szykman

et al. 2001) may lead to the development of tool translators that automate this process.

- **$FR^e \rightarrow F^i$ (process 1) and $F^e \rightarrow F^i$ (process 20):** Most tool support for the interpretation of object function is based on mechanisms of word recognition, given that many representations of function are described using natural language annotations. While there is a large number of general-purpose tools that provide interfaces for textual input (such as Word processors or electronic whiteboards), only few of them (e.g., the word generation system developed by de Vries et al. (2005)) offer more word-processing features than just editing. Future work on the interpretation of function can be expected to be driven by advances in both representing and reasoning about function, particularly in the area of design interoperability (Szykman et al. 2001).

4.4.2. Support for Re-Interpretation of Design Concepts

- **$SR^e \rightarrow S^i$ (process 3) and $S^e \rightarrow S^i$ (process 13):** Most research in re-interpretation has been done at the level of object structure. A system presented by Saund and Moran (1994) supports the creation of multiple interpretations of line drawings, by first decomposing and then reassembling elements of freehand drawings. The emerging shapes are then presented to the user for selection. A design agent capable of re-interpretation has been developed by Smith and Gero (2001) on the basis of Gero and Fujii's (2000) "push-pull" model of situated cognition. This system has been able to learn new shapes over sequences of action and (re-)interpretation that are themselves the result of the agent's modified experience.
- **$BR^e \rightarrow B^i$ (process 2) and $B^e \rightarrow B^i$ (process 19):** There has been no work to date on tool support for this process.
- **$FR^e \rightarrow F^i$ (process 1) and $F^e \rightarrow F^i$ (process 20):** There has been no work to date on tool support for this process.

4.5. COMPUTER-AIDED DESIGN SUPPORT FOR CONSTRUCTIVE MEMORY

Most work on computer-aided design tools includes support for memory in some way. There are two possible functions (F^t) related to this notion:

- to support retrieval of design concepts as stored
- to support re-construction of design concepts

Figure 8 highlights processes 4, 5 and 6 to represent constructive memory.

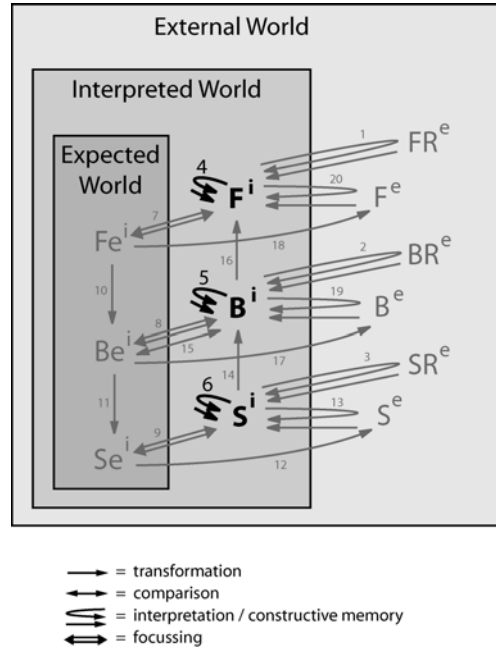


Figure 8. Constructive memory in the situated FBS framework

4.5.1. Support for Retrieval of Design Concepts as Stored

- **$S^i \rightarrow S^i$ (process 6):** Research in using memory of object structure includes work on feature-based modelling. A number of CAD systems provide design databases, repositories or libraries to store design features, such as pockets, holes and slots. Their reuse can lead to significant gains of productivity in designing. Techniques of feature extraction from geometrical CAD models can be viewed as another example of retrieving design concepts, although they require some additional computation. Here, features are implicitly stored in the pre-defined mappings underpinning common extraction techniques such as graph matching, syntactic pattern recognition and shape grammars (Shah 1991).
- **$B^i \rightarrow B^i$ (process 5):** Some recent work on design repositories has concentrated on including properties related to behaviour (B^0) and function (F^0) of the design object (Szykman et al. 2001; Mocko et al. 2004). In addition, approaches to capturing and reusing design rationale have focused on appropriate representations of previous object behaviour to be accessible for guiding the generation of new object structure (Chandrasekaran et al. 1993).

- **$F^i \rightarrow F^i$ (process 4):** Simple retrieval of object function is best exemplified by work on storing and reusing function (F^0) hierarchies in design repositories (Szykman et al. 2001) or case bases (Navinchandra et al. 1991). Approaches to retrieving implicitly stored functions include work, mentioned earlier, on inferring word relations based on WordNet (de Vries et al. 2005). Other work focuses on the construction of sub-functions using decomposition knowledge encoded in grammars (Sridharan and Campbell 2005).

4.5.2. Support for Re-Construction of Design Concepts

- **$S^i \rightarrow S^i$ (process 6), $B^i \rightarrow B^i$ (process 5) and $F^i \rightarrow F^i$ (process 4):** The idea of generating design concepts by situated re-construction rather than static retrieval from previous experience is quite new in design research. As a result, very little work has been done towards the development of computational models and tools that support this process. However, a number of research demonstrators have shown both the feasibility and the potential benefits of future constructive memory tools. Examples include neural network implementations used for the design of mechanical assemblies (Liew and Gero 2004), design optimisation (Peng and Gero 2006) and the exchange of product data between design tools (Kannengiesser and Gero 2007). The majority of this work provides support for re-construction of design concepts at all three levels, comprising function (F^0), behaviour (B^0) and structure (S^0).

5. Conclusion

Designing comprises a rich set of activities that is only beginning to be completely understood. Capturing these activities and defining them in a detailed framework is necessary to advance our understanding of design. The ontological framework presented in this chapter is a contribution to this aim. It extends our previous, object-centred work on representing the process of designing by adding a process-centred view. This view is based on the direct application of the FBS ontology to design activities, treating them as first-class entities with their own function, behaviour and structure, and no longer as mere derivatives of object-centred constructs. This provides a more structured description at a higher level of detail, which has the potential to make our framework of situated designing more amenable to other researchers.

We have shown that the process-centred ontology of designing also allows specifying a set of requirements for tool support. This is based on the connection we established between the FBS view of design processes and the FBS view of design tools. Specifically, tools are viewed as artefacts

whose functions (F^l) are specialised to supporting particular aspects of design processes, which themselves consist of combinations of function (F^p), behaviour (B^p) and structure (S^p). We have demonstrated how some of the outcomes of existing computer-aided design research and development can be mapped onto 20 classes of design processes represented in this way. One result of our mappings is that a lack of tool support can be identified for a number of design activities. At the level of granularity presented in this chapter, this concerns activities of re-interpretation and re-construction of design concepts, and reasoning and focussing on object function (F^o).

Our ontology allows understanding the research field of computer-aided design as the “materials science” of designing, concerned with creating and analysing tools to form appropriate “materials” of design processes at different levels of granularity. This is possible because the FBS ontology represents all design objects, tools and processes uniformly. Future research may use this ontology to create more fine-grained specifications of design tools. For example, different classes of feature extraction processes can be defined based on different classes of inputs (e.g., cubic, cylindrical or free-form shapes) and on different classes of transformations (e.g., graph matching, shape grammars, neural networks, etc.), and consequently different functions (F^l) of feature extraction tools can be derived. Information about specific process behaviour (B^p) and process function (F^p), on an instance level, can be added to derive more refined tool functions (F^l). Researchers and developers in computer-aided design can then identify specific gaps in the functions (F^l) of existing tools and generate the behaviour (B^l) and ultimately the structure (S^l) of new tools to close these gaps.

Acknowledgements

NICTA is funded by the Australian Government's Department of Communications, Information Technology and the Arts, and the Australian Research Council through Backing Australia's Ability and the ICT Research Centre of Excellence program.

References

- Asimov, M: 1962, *Introduction to Design*, Prentice-Hall, Englewood Cliffs.
- Bartlett, FC: 1932 reprinted in 1977, *Remembering: A Study in Experimental and Social Psychology*, Cambridge University Press, Cambridge.
- Bhatta, S, Goel, A and Prabhakar, S: 1994, Innovation in analogical design: A model-based approach, in JS Gero and F Sudweeks (eds) *Artificial Intelligence in Design '94*, Kluwer, Dordrecht, pp. 57-74.
- Bickhard, MH and Campbell, RL: 1996, Topologies of learning, *New Ideas in Psychology* **14**(2): 111-156.

- Campbell, MI, Cagan, J and Kotovsky, K: 1999, A-Design: An agent-based approach to conceptual design in a dynamic environment, *Research in Engineering Design* **11**(3): 172-192.
- Chandrasekaran, B, Goel, AK and Iwasaki, Y: 1993, Functional representation as design rationale, *IEEE Computer* **26**(1): 48-56.
- Chandrasekaran, B and Josephson, JR: 2000, Function in device representation, *Engineering with Computers* **16**(3-4): 162-177.
- Clancey, WJ: 1997, *Situated Cognition: On Human Knowledge and Computer Representations*, Cambridge University Press, Cambridge.
- Curtis, B, Kellner, MI and Over, J: 1992, Process modeling, *Communications of the ACM* **35**(9): 75-90.
- Deng, YM: 2002, Function and behavior representation in conceptual mechanical design, *Artificial Intelligence for Engineering Design, Analysis and Manufacturing* **16**(5): 343-362.
- Dewey, J: 1896 reprinted in 1981, The reflex arc concept in psychology, *Psychological Review* **3**: 357-370.
- Eastman, CM: 1999, *Building Product Models: Computer Environments Supporting Design and Construction*, CRC Press, Boca Raton.
- Gero, JS: 1990, Design prototypes: A knowledge representation schema for design, *AI Magazine* **11**(4): 26-36.
- Gero, JS and Fujii, H: 2000, A computational framework for concept formation for a situated design agent, *Knowledge-Based Systems* **13**(6): 361-368.
- Gero, JS and Kannengiesser, U: 2004, The situated function-behaviour-structure framework, *Design Studies* **25**(4): 373-391.
- Gero, JS and Kannengiesser, U: 2007, A function-behavior-structure ontology of processes, *Artificial Intelligence for Engineering Design, Analysis and Manufacturing* **21**(4), in press
- Gero, JS and Kazakov, V: 1998, Using analogy to extend the behaviour state space in creative design, in JS Gero and ML Maher (eds) *Computational Models of Creative Design IV*, Key Centre of Design Computing and Cognition, University of Sydney, Australia, pp. 113-143.
- Grecu, DL and Brown, DC: 1996, Learning by single function agents during spring design, in JS Gero and F Sudweeks (eds) *Artificial Intelligence in Design '96*, Kluwer, Dordrecht, pp. 409-428.
- Gross, MD: 1996, The electronic cocktail napkin – a computational environment for working with design diagrams, *Design Studies* **17**(1): 53-69.
- Guindon, R: 1990, Designing the design process: Exploiting opportunistic thoughts, *Human-Computer Interaction* **5**: 305-344.
- ISO: 2001, *Software Engineering – Product Quality – Part 1: Quality Model*, ISO/IEC 9126-1, International Organization for Standardization, Geneva, www.iso.ch
- Jacobsen, K, Sigurjonsson, J and Jacobsen, O: 1991, Formalized specification of functional requirements, *Design Studies* **12**(4): 221-224.
- Jozwiak, SF: 1987, Improving structural optimization programs using artificial intelligence concepts, *Engineering Optimization* **12**: 155-162.
- Jun, HJ and Gero, JS: 1997, Representation, re-representation and emergence in collaborative computer-aided design, in ML Maher, JS Gero and F Sudweeks (eds) *Preprints Formal Aspects of Collaborative Computer-Aided Design*, Key Centre of Design Computing and Cognition, University of Sydney, Australia, pp. 303-320.
- Kannengiesser, U and Gero, JS: 2006, Towards mass customized interoperability, *Computer-Aided Design* **38**(8): 920-936.

- Kannengiesser, U and Gero, JS: 2007, Agent-based interoperability without product model standards, *Computer-Aided Civil and Infrastructure Engineering* **22**(2): 80-97.
- Leclercq, P: 2001, Programming and assisted sketching, in B de Vries, JP van Leeuwen and HH Achten (eds) *CAAD Futures 2001*, Kluwer Academic Publishers, Dordrecht, pp. 15-32.
- Liew, P and Gero, JS: 2004, Constructive memory for situated design agents, *Artificial Intelligence for Engineering Design, Analysis and Manufacturing* **18**(2): 163-198.
- Mackenzie, CA and Gero, JS: 1987, Learning design rules from decisions and performances, *Artificial Intelligence in Engineering* **2**(1): 2-10.
- Maiden, NA and Sutcliffe, AG: 1992, Exploiting reusable specifications through analogy, *Communications of the ACM* **35**(4): 55-63.
- McNeill, T, Gero, JS and Warren, J: 1998, Understanding conceptual electronic design using protocol analysis, *Research in Engineering Design* **10**(3): 129-140.
- Miller, GA: 1995, WordNet: A lexical database for English, *Communications of the ACM* **38**(11): 39-41.
- Mocko, G, Malak, R, Paredis, C and Peak, R: 2004, A knowledge repository for behavioral models in engineering design, *Computers and Information Science in Engineering Conference '04*, Salt Lake City, UT.
- Navinchandra, D, Sycara, KP and Narasimhan, S: 1991, Behavioral synthesis in CADET, a case-based design tool, *IEEE Conference on Artificial Intelligence Applications*, Miami Beach, FL, pp. 217-221.
- Osterweil, LJ: 2005, Unifying microprocess and macroprocess research, in M Li, B Boehm and LJ Osterweil (eds) *Unifying the Software Process Spectrum*, Springer-Verlag, Berlin, pp. 68-74.
- Papalambros, P and Wilde, DJ: 2000, *Principles of Optimal Design: Modeling and Computation*, Cambridge University Press, Cambridge.
- Parmee I.C. (1996) Towards an optimal engineering design process using appropriate adaptive search strategies, *Journal of Engineering Design* **7**(4): 341-362.
- Peng, W and Gero, JS: 2006, Concept formation in a design optimization tool, in J van Leeuwen and H Timmermans (eds) *Innovations in Design Decision Support Systems in Architecture and Urban Planning*, Springer-Verlag, Berlin, pp. 293-308.
- Rescher, N: 2006, *Process Philosophical Deliberations*, Ontos-Verlag, Frankfurt.
- Reymen, IMMJ, Hammer, DK, Kroes, PA, van Aken, JE, Dorst, CH, Bax, MFT and Basten, T: 2006, A domain-independent descriptive design model and its application to structured reflection on design processes, *Research in Engineering Design* **16**(4): 147-173.
- Sacks, R, Eastman, CM and Lee, G: 2004, Parametric 3D modeling in building construction with examples from precast concrete, *Automation in Construction* **13**(3): 291-312.
- Saund, E and Moran, TP: 1994, A perceptually-supported sketch editor, *ACM Symposium on User Interface Software and Technology*, ACM Press, New York.
- Schön, DA: 1983, *The Reflective Practitioner: How Professionals Think in Action*, Harper Collins, New York.
- Schön, DA and Wiggins, G: 1992, Kinds of seeing and their functions in designing, *Design Studies* **13**(2): 135-156.
- Schwabacher, M, Ellman, T and Hirsh, H: 1998, Learning to set up numerical optimizations of engineering designs, *Artificial Intelligence for Engineering Design, Analysis and Manufacturing* **12**(2): 173-192.
- Shah, JJ: 1991, Assessment of features technology, *Computer-Aided Design* **23**(5): 331-343.
- Shah, JJ and Mäntylä, M: 1995, *Parametric and Feature-Based CAD/CAM: Concepts, Techniques, and Applications*, John Wiley & Sons, New York.

- Smith, GJ and Gero, JS: 2001, Interaction and experience: Situated agents and sketching, in JS Gero and FMT Brazier (eds) *Agents in Design 2002*, Key Centre of Design Computing and Cognition, University of Sydney, pp. 115-132.
- Smith, GJ and Gero, JS: 2005, What does an artificial design agent mean by being 'situated'?, *Design Studies* **26**(5): 535-561.
- Sridharan, P and Campbell, MI: 2005, A study on the grammatical construction of function structures, *Artificial Intelligence for Engineering Design, Analysis and Manufacturing* **19**(3): 139-160.
- Stone, RB and Wood, KL: 2000, Development of a functional basis for design, *Journal of Mechanical Design* **122**(4): 359-370.
- Suwa, M, Gero, JS and Purcell, T: 1999, Unexpected discoveries and s-inventions of design requirements: A key to creative designs, in JS Gero and ML Maher (eds) *Computational Models of Creative Design IV*, Key Centre of Design Computing and Cognition, University of Sydney, Sydney, Australia, pp. 297-320.
- Suwa, M and Tversky, B: 2002, External representations contribute to the dynamic construction of ideas, in M Hegarty, B Meyer and NH Narayanan (eds) *Diagrams 2002*, Springer-Verlag, Berlin, pp. 341-343.
- Szykman, S, Fenves, SJ, Keirouz, W and Shooter, SB: 2001, A foundation for interoperability in next-generation product development systems, *Computer-Aided Design* **33**(7): 545-559.
- Taggart, J: 1975, Sketching: An informal dialogue between designer and computer, in N Negroponte (ed.) *Reflections on Computer Aids to Design and Architecture*, Petrocelli Charter, New York, pp. 147-162.
- Umeda, Y, Ishii, M, Yoshioka, M, Shimomura, Y and Tomiyama, T: 1996, Supporting conceptual design based on the function-behavior-state modeler, *Artificial Intelligence for Engineering Design, Analysis and Manufacturing* **10**(4): 275-288.
- de Vries, B, Jessurun, J, Segers, N and Achten, H: 2005, Word graphs in architectural design, *Artificial Intelligence for Engineering Design, Analysis and Manufacturing* **19**(4): 277-288.
- Weisberg, DE: 2000, The electronic push, *Mechanical Engineering* **122**(4): 52-59.
- Ziemke, T: 1999, Rethinking grounding, in A Riegler, M Peschl and A von Stein (eds) *Understanding Representation in the Cognitive Sciences: Does Representation Need Reality?*, Plenum Press, New York, pp. 177-190.