Artificial Intelligence for Engineering Design, Analysis and Manufacturing (2007), *21*, 379–391. Printed in the USA.

DOI: 10.1017/S0890060407000340

A function–behavior–structure ontology of processes

JOHN S. GERO[1] AND UDO KANNENGIESSER[2]

[1]Krasnow Institute for Advanced Study and Volgenau School of Information Technology and Engineering, George Mason University, Fairfax, Virginia, USA
[2]NICTA, Alexandria, Australia

(RECEIVED July 2006; ACCEPTED May 2007)

Abstract

This paper presents how the function–behavior–structure (FBS) ontology can be used to represent processes despite its original focus on representing objects. The FBS ontology provides a uniform framework for classifying processes, and includes higher level semantics in their representation. We show that this ontology supports a situated view of processes based on a model of three interacting worlds. The situated FBS framework is then used to describe the situated design of processes.

Keywords: Function–Behavior–Structure Framework; Process Ontology; Situatedness

1. INTRODUCTION

Ontologies are structured conceptualizations of a domain in terms of a set of entities in that domain and their relationships. They provide uniform frameworks to identify differences and similarities that would otherwise be obscured. In the design domain, a number of ontologies have been developed to represent objects, specifically artifacts (Chandrasekaran & Josephson, 2000; Stone & Wood, 2000; Kitamura et al., 2004; International Alliance for Interoperability, 2006). They form the basis for a common understanding and terminological agreement on all relevant properties of a specific artifact or class of artifacts. Ontologies can then be used to represent the evolving states of designing these artifacts or as knowledge representation schemas for systems that support designing.

Design research is a field that has traditionally shown particular interest in explicit representations of processes besides objects. A number of process taxonomies have been created that classify different design methods (e.g., Cross, 1994; Hubka & Eder, 1996). However, most of this work has not been based on process ontologies, which makes comparison of the different taxonomies difficult. Ontologies are richer than taxonomic class hierarchies, as they provide definitions and constraints for an entity's properties and relationships. Some of the efforts towards stronger ontological foundations for process representation have been driven by the need to effectively plan and control design and construction processes. For example, recent work on four-dimensional CAD systems links three-dimensional object models to project schedules (Haymaker & Fischer, 2001).

A large number of process ontologies and representations have been developed, with varying degrees of domain or task specificity. For example, IDEF0 [National Institute of Standards and Technology (NIST), 1993] is a high-level ontology for modeling industry processes at any level of detail, distinguishing between input, control, output, and mechanism. Another, more recent high-level ontology is PSL (NIST, 2000). PERT (Wiest & Levy, 1977) is a process representation primarily used for scheduling tasks in projects. The Quirk Model (Motus & Rodd, 1994) describes computational processes and their timing constraints to enable analysis and control of overall system speed.

Most process ontologies and representations have a view of processes that is based on flows of activities and/or sequences of states. Semantics, capturing the processes' applicability in a purposive context, are generally not included in most process ontologies. Such semantics are needed to guide the generation, analysis, and evaluation of a variety of processes. As research increasingly focuses on automating parts of the selection or synthesis of processes, existing process ontologies provide inadequate representations for computational support.

An ontology that supports semantics is based on the function–behavior–structure (FBS) framework introduced by Gero (1990) that later became an ontology (Gero & Kannengiesser, 2004). Its original focus was on representing artificial objects. In this paper we show how this focus can be extended to include processes. Our contribution thus

Reprint requests to: Udo Kannengiesser, NICTA, Locked Bag 9013, Alexandria NSW 1435, Australia. E-mail: udo.kannengiesser@nicta.com.au

consists of a novel interpretation and application of an existing ontology rather than a novel ontology of processes itself. Section 2 describes the basics of this approach and demonstrates how the FBS ontology can be used to classify processes. Section 3 develops a situated view of processes, which accounts for situation-specific changes in process representations at three levels: function, behavior, and structure. Section 4 presents a framework of situated process design using a design optimization process as an example. Section 5 concludes the paper.

2. THE FBS ONTOLOGY

2.1. The FBS view of objects

The FBS ontology provides three high-level categories for the properties of an object:

1. The *function* of an object is defined as its teleology ("what the object is for"), which is largely domain dependent.
2. The *behavior* of an object is defined as the attributes that can be derived from its structure ("what the object does"). Most instances of behavior are domain dependent.
3. The *structure* of an object is defined as its components and their relationships ("what the object consists of"). The structure of most objects can be described in terms of geometry, topology, and material.

Humans construct connections between function, behavior, and structure through experience and through the development of causal models based on interactions with the object. Specifically, function is ascribed to behavior by establishing a teleological connection between the human's goals and observable or measurable effects of the object. Behavior is causally connected to structure, that is, it can be derived from structure using physical laws or heuristics. There is no direct connection between function and structure (de Kleer & Brown, 1984).

The generality of the FBS ontology allows for multiple views of the same object. This enables the construction of different models depending on their purpose. For example, an architectural view of a building object includes different FBS properties than a structural engineering view. This is most striking for the building's structure: architects typically view this structure as a configuration of spaces, whereas engineers often prefer a disjoint view based on floors and walls.

Multiple views can also be constructed depending on the required level of aggregation. This allows modeling objects as assemblies composed of subassemblies and individual parts. Each of these components can again contain other subassemblies or parts. No matter which level of aggregation is required, the FBS ontology can always be applied.

2.2. The FBS view of processes

Objects and processes have traditionally been regarded as two orthogonal views of the world. The difference between these views is primarily based on the different levels of abstraction involved in describing what makes up their structure. The structure of physical or virtual objects consists of representations of material, geometry, and topology. These representations can be easily visualized and understood. Processes are more abstract constructs that include transitions from one state of affairs to another.

The high-level categorizations provided by the FBS ontology create an integrative view that treats objects and processes in a uniform manner. This is possible because the FBS ontology does not include the notion of time. Although on an instance level this notion is fundamental to the common distinction between objects and processes, on an ontological level there is no time-based difference between them. All states of any entity at any point in time can be described by a set of properties that can be classified as function, behavior, and structure.

The notion of function applies to any entity as it only accounts for the observer's goals, independent of the entity's embodiment as an object or as a process. We give examples of process functions in the following sections.

Behavior relates to those attributes of an entity that allow comparison on a performance level rather than on a compositional level. Such performance attributes are representations of the effects of the entity's interactions with its environment. Typical behaviors of processes are speed, rate of convergence, cost, amount of space required, and accuracy.

Although process function and process behavior are not fundamentally different to object function and object behavior, process structure is clearly distinctive. It includes three components and two relationships (Fig. 1). 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).

2.2.1. *Input and output*

The input and the output structure elements represent properties of other entities in terms of their variables and/or their values. For example, the process of transportation changes only

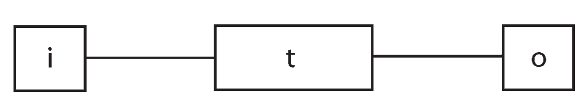

Fig. 1. The structure of a process; i, input; t, transformation; o, output.

the values for the location of a (physical) object (e.g., the values of its x, y, and z coordinates). As the input and output contain the same variables here, such a process can be characterized as homogenous. Heterogenous processes, in contrast, use disparate variables as input and output. For example, the process of electricity generation takes mechanical motion as input and produces electrical energy as output.

Input and output may refer not only to (properties of) objects but also to (properties of) other processes. For example, it is not uncommon for software procedures to accept the output of other procedures as their input or to return procedure calls as their output. All variables and values used as input and output of a process may refer to the function, behavior, or structure of other objects or processes.

2.2.2. Transformation

A common way to describe the transformation 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 a Unified Modeling Language (UML) activity diagram. Such descriptions are often viewed as a collection of subordinate processes. In the software example, this is most explicit when a procedure calls other procedures that are possibly located in other program components or other computers. Every subprocess can again be modeled in terms of function, behavior, and structure.

2.2.3. Relationships

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

2.2.4. Process classifications based on the FBS ontology

The FBS view of processes provides a means to classify different instances of design processes according to differences in their function, behavior, or structure. (To avoid confusion, we will use the terms "object function," "object behavior," and "object structure" whenever we refer to objects. In all other cases, the default assumption will be that function, behavior, and structure refer to processes.) Take Gero's (1990) eight fundamental classes of processes involved in designing; they can be distinguished by differences in their input and output. For example, whereas synthesis is a transformation of expected object behavior (i) into object structure (o), analysis transforms object structure (i) into object behavior (o). Within each of these fundamental processes we can identify different instances if we reduce the level of abstraction at which input and output are specified. For example, different instances of the process class of analysis can be defined based on the specific kind of output they produce: stress analysis computes stress (o), thermal analysis computes temperature (o), cost analysis computes cost (o), and so forth. Other process instances can be based on the transformation. For example, the synthesis of a design object can be carried out using a range of different transformations or techniques to map expected behavior onto structure. Examples include case-based reasoning, genetic algorithms, or gradient-based search methods.

Other process classifications and taxonomies are similarly based on differences in structure. For example, Hubka and Eder (1996) distinguish between six subprocesses of designing, each of which specifies distinct abstraction levels describing their input and output. Processes can also be distinguished according to their behavior and function. For example, design optimization processes can be characterized on the basis of differences in their speed, differences in the amount of space they require, or other behaviors. Another example has been provided by Sim and Duffy (1998), who propose a multidimensional classification of machine learning processes in design that can be mapped on structure and function of a process. Specifically, learning processes are grouped according to input knowledge and learning trigger (both i), knowledge transformers (t), output knowledge (o) and learning goal (F).

3. SITUATED FBS REPRESENTATIONS OF PROCESSES

3.1. Situatedness

Designing is an activity during which designers perform actions 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 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 & Wiggins, 1992). This interaction between the designer and the environment strongly determines the course of designing. This idea is called "situatedness," the foundational concepts of which 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."

Gero and Fujii (2000) have developed a framework for situated cognition using agents, which describes the designer's interpretation of their environment as interconnected sensation, perception, and conception processes. Each of them consists of two parallel processes that interact with each other: a *push process* (or data-driven process), where the production of an internal representation is driven ("pushed") by the environment, and a *pull process* (or expectation-driven process), where the interpretation is driven ("pulled") by some of the designer's current concepts, which has the effect that the interpreted environment is biased to match the current expectations.

The environment that is interpreted can be external or internal to the agent. The situated interpretation of the internal environment accounts for the notion of constructive memory. Gero (1999) showed the relevance of this notion in the area of design research. Constructive memory is best exemplified by a paraphrase of Dewey by Clancey (1997): "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, 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, whether carried out by humans or a design system, uses first-person knowledge grounded in the designer's interactions with their environment (Bickhard & Campbell, 1996; Clancey, 1997; Ziemke, 1999; Smith & 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, and 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.

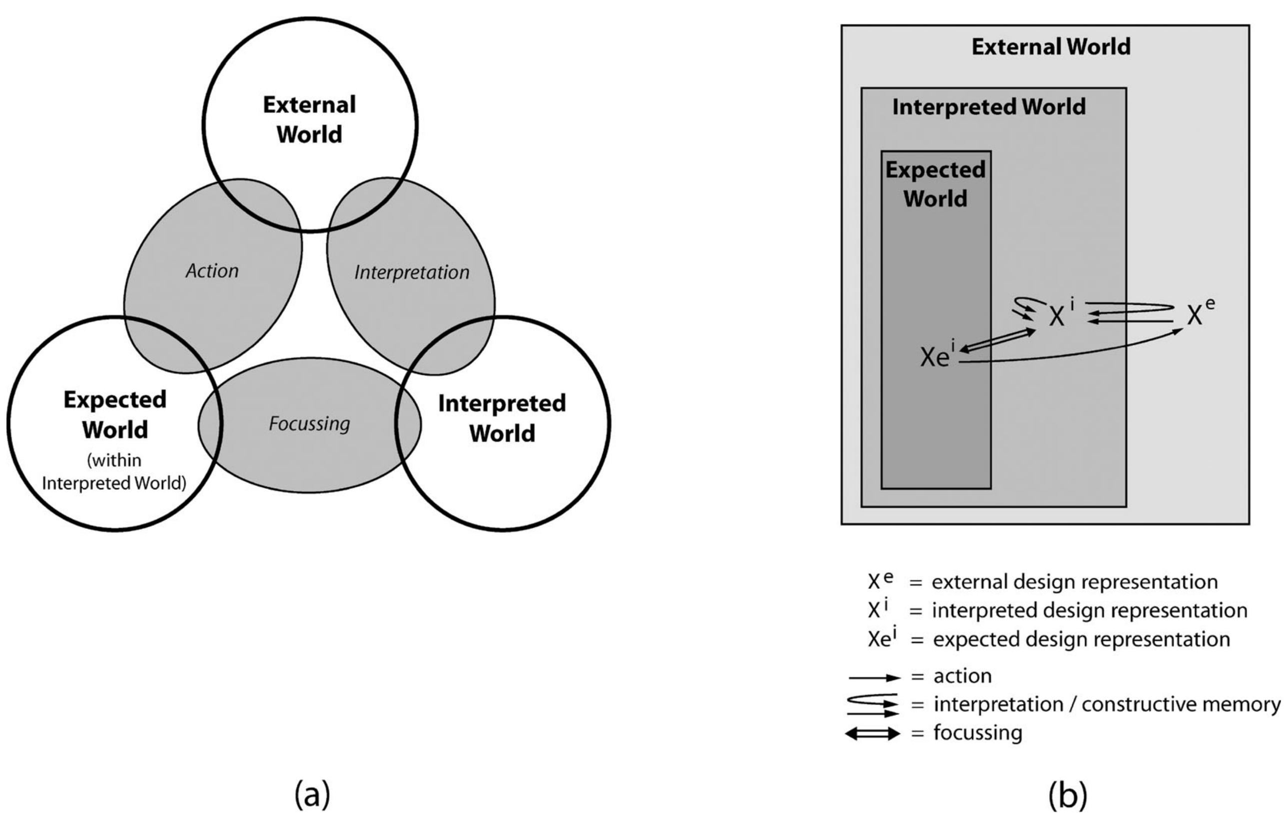

Fig. 2. Situatedness as the interaction of three worlds: (a) a general model and (b) a specialized model for design representations (Gero & Kannengiesser, 2004).

Gero and Kannengiesser (2004) have modeled situatedness as the interaction of three worlds, each of which can bring about changes in any of the other worlds (Fig. 2a).

1. The *external world* is the world that is composed of representations outside the designer or design system.
2. The *interpreted world* is the world that consists of the sensory experiences (percepts) and concepts of the designer or design system. It is the internal representation of that part of the external world that the designer or design system interacts with.
3. The *expected world* is the world imagined actions will produce. It is the environment in which the designer or design system predicts the effects of actions according to the current goals and interpretations. The term "expected world" extends the notion of simple desirability by incorporating goals that serve as "expected," realistic benchmarks for actions.

These three worlds are linked together by three classes of activities. *Interpretation* transforms variables that are sensed in the external world into the interpretations of sensory experiences, percepts, and concepts that compose the interpreted world. *Focusing* takes some aspects of the interpreted world and uses them as goals for the expected world that then become the basis for the suggestion of actions. These actions are expected to produce states in the external world that reach the goals. *Action* is an effect that brings about a change in the external world according to the goals in the expected world.

Figure 2b presents a specialized form of this view with the designer or design system (as the internal world) located within the external world and placing general classes of design representations into the resultant nested model. The set of expected design representations (Xe^i) corresponds to the notion of a design state space. 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 reinterpretation, changing the interpreted world. Figure 2b represents both interpretation and constructive memory as "push–pull" processes, as outlined earlier in this section.

The view of three worlds captures the idea that multiple (interpreted) views can be constructed from the same external world, and that multiple goals can arise from different views. Gaps between current goals and current (interpreted) views of the world then lead to individual actions aiming to reduce these gaps.

3.2. Constructing multiple views for multiple purposes

Gero and Kannengiesser's (2004) three-world model can be used to construct situated FBS representations of

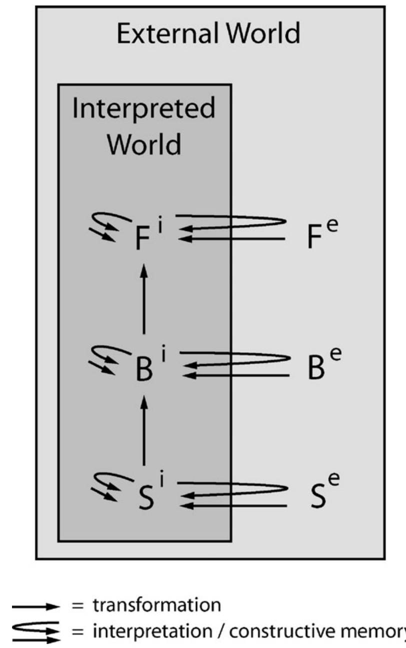

Fig. 3. External and interpreted function–behavior–structure representations of processes.

processes.[1] The main basis for creating a situated view is the distinction between the external and the interpreted world. Locating function, behavior, and structure of a process in each of these worlds (Fig. 3), results in six ontological categories:

1. external function (F^e),
2. external behavior (B^e),
3. external structure (S^e),
4. interpreted function (F^i),
5. interpreted behavior (B^i), and
6. interpreted structure (S^i).

Process representations of categories 4, 5, and 6 are generated via push–pull activities involving only the internal world (constructive memory) or both internal and external worlds (interpretation). In addition, B^i can be generated by transforming S^i and F^i can be generated by transforming B^i.

3.2.1. External versus interpreted structure of a process

Most design ontologies cannot deal with different interpretations of a process, as they do not distinguish between external and interpreted worlds. Such interpretations are often required for representing process structure. This is because of a number of reasons.

Many instances of process S^e are transient and time based. Delineating the components of the process (i.e., input, transformation, and output) from one another as well as from other

[1] Although it is clearly the processes that are situated, in this paper we use the term "situated FBS representations of processes" as shorthand for "FBS representations of situated processes."

entities in the external world then requires acts of discretization from continuous flows of events according to the observer's current knowledge and goals. For example, it is possible to view the intermediate results of an iterative process as part of its transformation or, alternatively, as part of its output.

The kind of components of the process structure and the level of detail used to describe them are similarly dependent on the stance of the observer. One example, already mentioned in Section 2.2.2, is the range of possible views of the transformation from a detailed procedural plan to an object or a simple "black box." There are also many examples for disparate views of the input and output of the same process. Take a pressing process in the automotive industry: a manufacturing engineer generally views the input and the output of this process in terms of geometry of the sheet steel to be transformed. In contrast, a costing expert typically views the input and output of the same process in terms of (material, labor, etc.) cost and yield, respectively. Similar view-dependent examples have been presented by NIST (2004).

3.2.2. External versus interpreted behavior of a process

The distinction between external and interpreted worlds is also useful when dealing with the performance or behavior of a process. This allows different observers to reason about different performance aspects of a process according to the current situation. For example, the cost of burning fuel might be important for the owner of a car; however, this cost is usually not directly relevant for the hitchhiker sitting on their passenger seat. Another example is the amount of memory space needed by a particular computational process. This behavior is usually worth considering for users only if their hardware resources are limited for current purposes. The kind of B^i constructed is largely influenced by individual experience, and must therefore be clearly distinguished from B^e.

The kind of B^i that an observer is interested in also affects the way in which that observer interprets the structure that is responsible for causing that behavior. This is the case when no B^e and no memories of previous B^i are available, and the B^i must be derived from S^i. If, for instance, the speed of a process is to be measured, then a structural description of the input and output of that process must be produced that contains references to some quantities and time units. If the amount of space required by the process is to be measured, then there must be a structural description that provides sufficient detail about the path of transformation for given inputs and outputs.

3.2.3. External versus interpreted function of a process

The need to separate the interpreted from the external world is most obvious for the function of a process. Individual observers have the autonomy to interpret function according to their own goals and desires that are likely to differ from others. They may come up with various processes F^i, which may be independent of the constraints imposed by process structure and behavior. For example, it is solely dependent on an observer's previous experience or current goals if they ascribe the function "operate time efficiently" to a manufacturing process, even though the exact speed of that process (as its B^i) or an explicit F^e may be given.

3.3. Constructing multiple purposes from multiple views

Let us add the expected world to the interpreted and external world (Fig. 4). The number of ontological categories now increases to nine:

1. F^e,
2. B^e,
3. S^e,
4. F^i,
5. B^i,
6. S^i,
7. expected function (Fe^i)
8. expected behavior (Be^i), and
9. expected structure (Se^i).

Actions that aim to reduce gaps between the interpreted and the expected world involve the creation or modification of S^e. A number of examples of such actions will be presented throughout this section. The B^i derived from the structure resulting from an action can be assumed to reflect "real" performance, and the functions (F^i) ascribed to those behaviors can

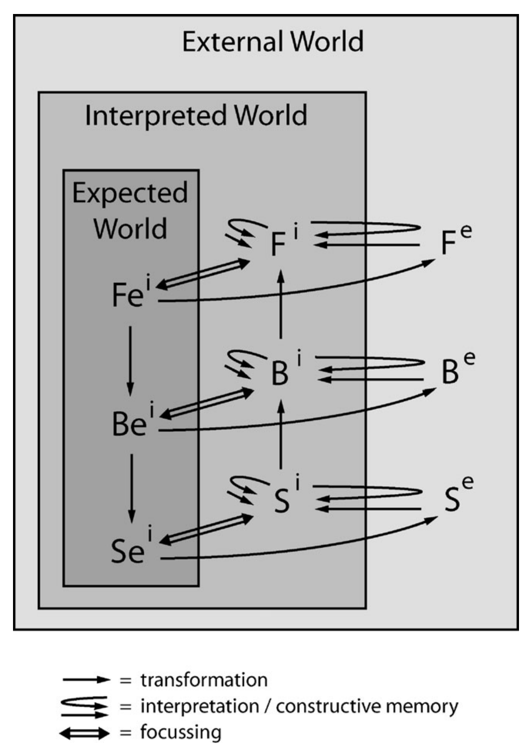

Fig. 4. External, interpreted, and expected function–behavior–structure representations of processes.

be assumed to reflect "real" purposes. Actions that directly create or modify B^e or function (F^e) are not grounded in the "real" world. They capture representations used for communication about the "real" world rather than the "real" world itself.

3.3.1. Interpreted versus expected structure of a process

Expected process structure describes the composition of desired processes. Actions can then be performed to realize (represent) the desired processes in the external world. Processes established by these actions are often called strategies, realized either by individuals (Gruber, 1989) or by organizations (Chandler, 1962).

The interaction between the external, interpreted, and expected structure of strategies is an instance of Schön's (1983) concept of "reflection in action." It allows for reflective reasoning about one's interactions with the external world, which has the potential of substantially changing current strategies (Hori, 2000). Work in management science has established the term "strategizing" to denote the interactive construction of new strategies by cycles of interpretation and action (Cummings & Wilson, 2003). Strategizing combines the traditional idea of top-down implementation of preformed strategies with more recent models of bottom-up recognition of new strategies as "patterns in a stream of actions" (Mintzberg & Waters, 1985).

3.3.2. Interpreted versus expected behavior of a process

Differences between the interpreted and the expected world at the level of the behavior of a process are what project managers have to deal with. They represent gaps between the actual (interpreted) and the desired (expected) state of a process in terms of performance. Common examples include the speed, cost, and accuracy of a process that may diverge from the corresponding target values specified in the project plan. There are two possibilities to reduce or eliminate the gap between the interpreted and the Be^i of the process. First, the Be^i may be adjusted to the current state of the process to satisfy the project plan. Second, corrective action may be taken to change the external world such that the interpreted performance (B^i) better matches the current expectations. This involves transforming the Be^i into S^e via Se^i, transforming that external structure into interpreted structure (S^i), and, finally, deriving B^i. We refer to this set of activities as a composite, "performance-oriented" activity. It differs from the elementary transformation of Be^i into B^e, which may be described as a "communicative action about performance." Examples of this communicative action include justifying the selection of a particular design strategy (Clibbon & Edmonds, 1996) and documenting design rationale to explain decisions taken in a design process (Chandrasekaran et al., 1993).

3.3.3. Interpreted versus expected function of a process

The gap between a potential purpose and the currently focused purpose ascribed to the process can be determined using the distinction between interpreted and expected function of a process. Similar to behavior, this gap may be reduced or eliminated through adoption of new Fe^i or through action to modify the external world. The latter requires transforming the Fe^i into Be^i, and then follows the "performance-oriented" activity (described in Section 3.3.2) to obtain an B^i to which, finally, a new F^i is ascribed. We refer to this set of activities as a composite, "goal-oriented" activity. It differs from the elementary transformation of Fe^i into F^e, which may be described as a "communicative action about goals." In other words, a goal-oriented activity decides on a process structure that exhibits a set of performance criteria that are expected to achieve the process goals. Examples for expectations related to process goals constraining the selection of design strategies have been articulated by von der Weth (1999) to include "carefulness" and "thoughtfulness," depending on the degree of complexity, novelty, and dynamism of a given situation.

4. SITUATED DESIGN OF PROCESSES

The situated FBS representations presented in Section 3 and the activities connecting them provide a basic understanding of processes from a situated perspective. In this section, we extend the application of the ontology to explore the notion of situated design of processes. Although the notion of process design is well known in fields such as business process reengineering, manufacturing planning, and strategic management, the role of situatedness in designing processes has received little attention to date. Section 4.1 presents a situated framework of design from the basics described in this paper. Section 4.2 illustrates the situated design of an optimization process.

4.1. The situated FBS framework

Gero and Kannengiesser's (2004) situated FBS framework (Fig. 5) can be seen as an extension of the process descriptions presented in Section 3. It contains 20 activities that include two additional classes of activities with respect to those presented earlier in this paper: first, the framework represents external requirements related to the function (FR^e), behavior (BR^e), and structure (SR^e) of processes. External requirements are given to the process designer via communicative actions from a customer or another authority, internal, or external to the organization of the designer. Designing typically starts with these requirements, and additional external requirements are often given later in the process of design. Second, the framework adds the activity of comparison between Be^i and B^i. This is seen as an important activity in most models of design. It serves as a shortcut for evaluating if the goals of a design have been achieved by identifying gaps between expected and interpreted worlds at the behavior level rather than the function level.

Designing is closely related to the goal-oriented activity in Section 3.3.3. It aims to create or change structure in the external world (S^e) so that the gap between expected and

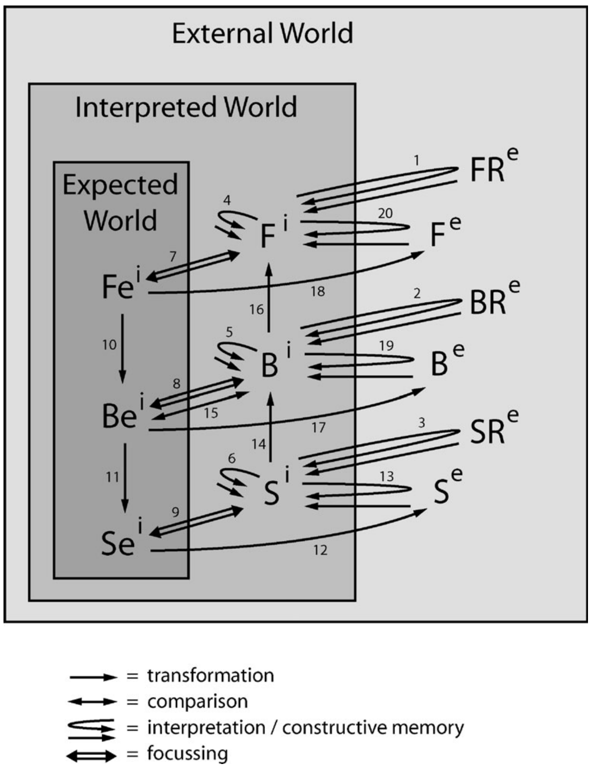

Fig. 5. The situated FBS framework (Gero & Kannengiesser, 2004).

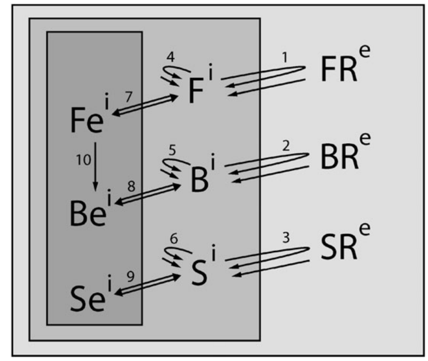

Fig. 6. Formulation.

interpreted function is reduced or eliminated, via the "shortcut" evaluation at the behavior level. Gero (1990) has presented a detailed description of eight fundamental steps in designing, which Gero and Kannengiesser (2004) have mapped onto the 20 activities in the situated FBS framework. This description is independent of the domain of designing, and can include objects as well as processes as the object being designed.

1. *Formulation* defines the design task by delineating a state space of potential design solutions (termed the structure state space) and a set of criteria for assessing these solutions (termed the behavior state space). This activity uses a set of goals (termed the function state space) and constraints that are given to the designer by external specification or are constructed based on the designer's own experience. In the situated FBS framework, this design step is composed of activities 1–10 (all numbers in the framework are only labels and do not imply any order of execution; Fig. 6).
2. *Synthesis* produces a design solution in terms of a point in the structure state space. In the situated FBS framework, this design step is composed of activities 11 and 12 (Fig. 7).
3. *Analysis* derives the behavior from the design solution. In the situated FBS framework, this design step is composed of activities 13 and 14 (Fig. 8).
4. *Evaluation* assesses the design solution on the basis of the formulated criteria, that is, by comparison of the behavior derived from the design solution and the expected behavior. In the situated FBS framework, this design step is activity 15 (Fig. 9).
5. *Documentation* produces an external representation of the final design solution for purposes of communicating that solution. In the situated FBS framework, this design step is composed of activities 12, 17, and 18 (Fig. 10).
6. *Reformulation type 1* redefines the structure state space. This may or may not entail redefining the behavior state space. In the situated FBS framework, this design step is activity 9, with activities 3, 6, and 13 as potential drivers (Fig. 11).
7. *Reformulation type 2* redefines the behavior state space. This may or may not entail redefining the structure state space and function state space. In the situated FBS framework, this design step is activity 8, with activities 2, 5, 14, and 19 as potential drivers (Fig. 12).
8. *Reformulation type 3* redefines the function state space. This may or may not entail redefining the behavior state space. In the situated FBS framework, this design step is activity 7, with activities 1, 4, 16, and 20 as potential drivers (Fig. 13).

The numbering of the eight design steps (analogous to the 20 labeled activities) does not prescribe a fixed order of execution. Although it is often expected for routine design tasks

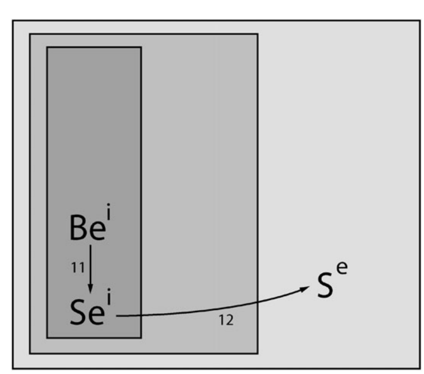

Fig. 7. Synthesis.

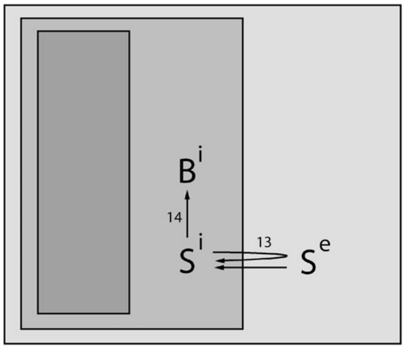

Fig. 8. Analysis.

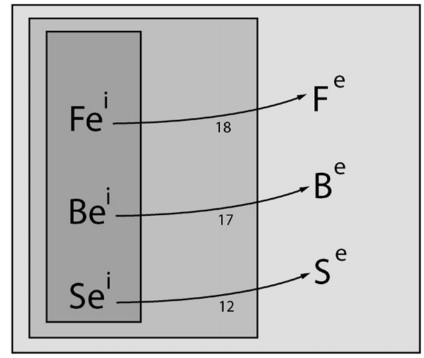

Fig. 10. Documentation.

to involve only a top-down, linear execution along a schema of "formulation–synthesis–analysis–evaluation– documentation," all three types of reformulation are frequent, transforming most design processes into a mixture of top-down and bottom-up reasoning.

4.2. Example: Situated design of an optimization process

To illustrate the steps and activities involved in situated process design, we use the example of design optimization. Optimization can be described as a process that takes as its input a problem statement, including a set of object design parameters, required object performances and object constraints. Its output includes a set of values for the object design parameters, representing the best performing object design solution. The transformation is commonly viewed as encompassing the activities shown within the UML activity diagram in Figure 14. The loop in the diagram accounts for the iterative structure that is common in most optimization processes.

Each activity can be regarded as an individual process. For example, the activity "produce a mathematical model" is a subprocess of optimization, which can again be viewed in terms of FBS. A distinctive function of this subprocess is to provide the necessary formalism for applying computational operations on the optimization problem. Behaviors include performance characteristics (such as accuracy and speed) that support the function of the subprocess within optimization. Structure includes a sequence of activities as part of the transformation component (Fig. 15).

The activities of this subprocess may be viewed in ever more detail, and multiple FBS models can be constructed and organized in a hierarchical structure. This is similar to object-centered views of the world, in which objects can be modeled as assemblies of other objects at multiple layers.

Carrying out an optimization process can be viewed as the act of designing its structure. In the situated FBS framework, this can be modeled as follows:

1. *Formulation* includes the interpretation of external requirements (FR^e, BR^e, and SR^e) and the generation of additional, "implicit" requirements (F^i, B^i, and S^i) via constructive memory. External requirements are often given to the designer in form of (or in conjunction with) the problem statement. For example, FR^e may be stated "to support the conceptual stage in product development," BR^e may specify time constraints (i.e., required speed) on the optimization process, and SR^e includes the problem statement. The interpretation of these requirements is subjective to the individual designer. Implicit requirements may include implicit assumptions about resource efficiency (F^i) in terms of computational tools and human labor, cost considerations

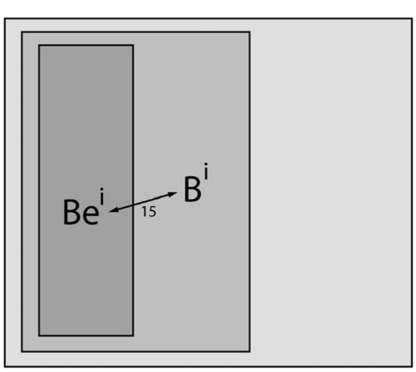

Fig. 9. Evaluation.

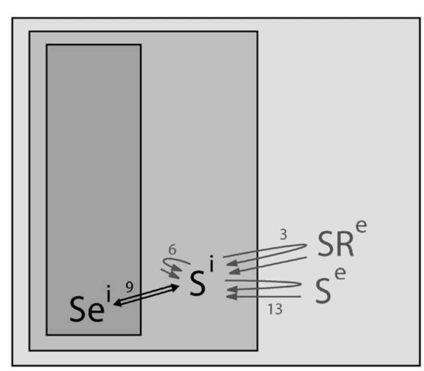

Fig. 11. Reformulation type 1 is depicted in black and the activities representing potential drivers for this design step are depicted in gray.

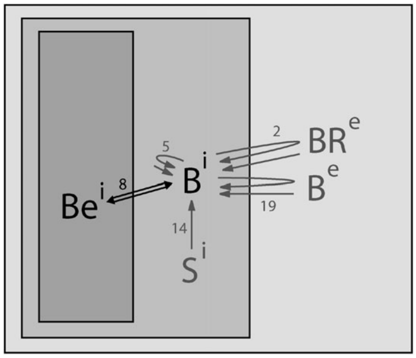

Fig. 12. Reformulation type 2 is depicted in black and the activities representing potential drivers for this design step are depicted in gray.

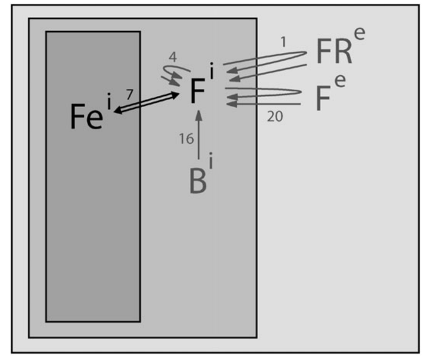

Fig. 13. Reformulation type 3 is depicted in black and the activities representing potential drivers for this design step are depicted in gray.

(B^i), and the process structure (S^i) interpreted from Figure 14. Implicit requirements also include refinements of these concepts in form of FBS views of the subprocesses of optimization, such as outlined for the subprocess "produce a mathematical model." The construction of implicit requirements is heavily based on individual experience. The design state space of the optimization process is formulated by focusing on the explicit and implicit process requirements, and by constructing additional Be^i based on Fe^i. An example for the latter is the designer's decision about the expected accuracy (Be^i) of the optimization process. Here, given that this process is "to support the conceptual stage in product development" (Fe^i), the designer may select a lower accuracy than would be needed if the optimization was to support more detailed stages in product development.

2. *Synthesis* instantiates and externalizes the structure of formulated processes and subprocesses. Take the

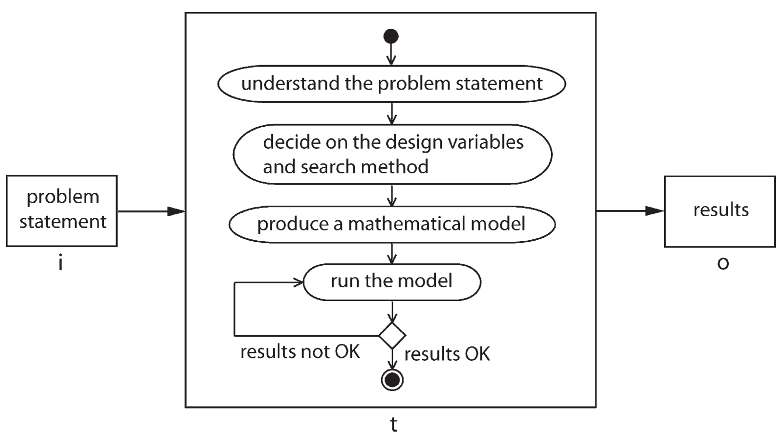

Fig. 14. The process structure of design optimization.

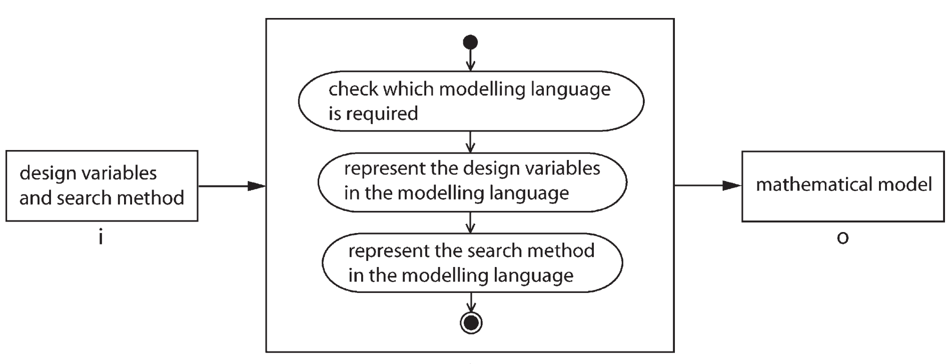

Fig. 15. The process structure of "produce a mathematical model."

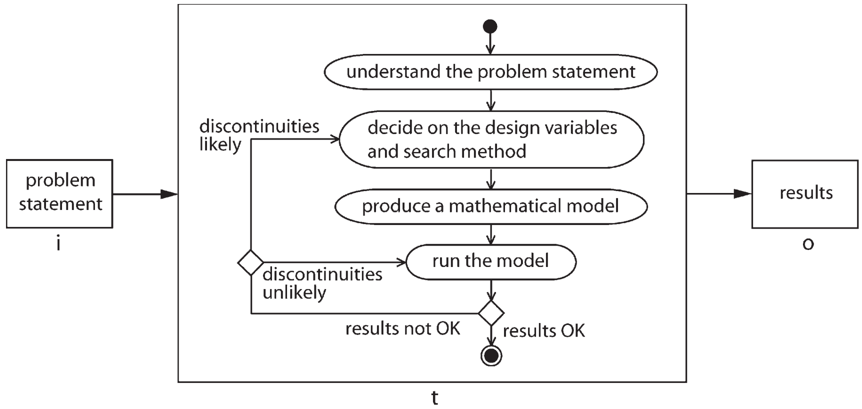

Fig. 16. The reformulated process structure of a design optimization process through the addition of a new iteration.

overall structure shown in Figure 14; synthesis includes determining not only the final results, but also the specific object design variables and search methods, and the specific path on which the optimization proceeds (e.g., if and how many times it iterates).

3. *Analysis* involves interpreting the structure of the processes that have been externalized through synthesis. What the outcomes of this activity are depends on the experience and current goals of the designer. For example, the designer may choose to look closer at the output rather than the transformation of the optimization, if accuracy is given priority over speed for evaluating process performance.
4. *Evaluation* compares the actual performance of optimization against the expected performance.
5. *Documentation* externalizes the representation of the final process design for purposes of communication. This includes the capture of process-centered design rationale for explaining why a particular optimization method was chosen.
6. *Reformulation type 1* reformulates optimization structure, which is usually done when performance is unsatisfactory. Common examples of reformulating optimization structure are the elimination of object design variables and the modification of search methods. They are necessary in many nonlinear optimization problems where discontinuities in the search space produce difficulties for gradient-based search methods in finding the global optimum result. Figure 16 shows the example of a reformulated optimization structure through modification of object design variables or search method. What exactly this modification produces depends on the individual experience of the designer, which is captured as interpretation and constructive memory in the situated FBS framework.
7. *Reformulation type 2* reformulates optimization behavior. This is frequently driven by new project constraints, represented as external requirements on behavior (BR^e). For example, the project manager may demand a faster pace of optimization because of increased market competition.
8. *Reformulation type 3* reformulates optimization function. Typically, this is driven by external requirements (FR^e) that define a new role for the optimization in the product development process. For example, they may change the function "to support the detail design stage," which may then lead to changed expected behaviors such as increased accuracy.

5. CONCLUSION

The FBS ontology as a structured conceptualization of the domain of processes was presented. We claim that any class of process can be represented using this ontology. A number of examples of processes in the design domain were described in this paper demonstrating its coverage. Future work will focus on providing a more systematic evaluation of the ontology across different domains. Our ontology provides a uniform representation that allows locating as well as distinguishing between them. This enables comparison of different process models, even if they use differing terminologies or notations. Efforts towards unifying models of the design process (Grabowski et al., 1998) may benefit from this work.

Integrating function and behavior in a process ontology adds semantics to process representations, which accounts for their applicability in a purposive context. This is useful for knowledge representations of processes, as they can be deployed by a knowledge-based system to generate, compare, and execute specific processes according to its current goals. Such knowledge representations are equivalent to Gero's (1990) design prototypes based on the FBS ontology for design objects. The ability to support different views and purposes of processes at functional, behavioral, and structural levels increases flexibility and applicability of the system in different situations. The situated FBS framework has been presented to model the steps and activities involved in the situated design of processes.

There have been other approaches that use similar notions of function, behavior, and structure (Stroulia & Goel, 1995; Murdock & Goel, 2001). However, their focus is on modeling reflective reasoning architectures rather than on specifying a general process ontology. We can see the opportunity to deploy our FBS ontology within a reasoning mechanism, exploiting its ability to use the same fundamental constructs—function, behavior, and structure—as for objects. This allows developing design systems or agents that can flexibly reason about a variety of objects and processes without having to implement different, specialized cognitive mechanisms. As everything in the world looks the same when viewed in terms of FBS, only one cognitive mechanism is required.

6. QUESTIONS AND ANSWERS

6.1. Question 1

"Please describe where the situatedness lies in your model."

6.2. Answer 1

"The fundamental concepts of situatedness [described in Section 3.1 of this article] are integrated into our model in the following ways:

- *Interaction between making and seeing* [first paragraph in Section 3.1]: This concept is represented by the interaction of the three worlds, which is used as the foundation of the situated FBS framework. The three connections between these worlds, referred to as interpretation, focusing, and action, are explicitly represented as classes of design activities in Figure 5.
- *Push and pull processes describing the designer's interpretation of the environment* [third paragraph in Section 3.1]: This concept is directly integrated into our framework by the explicit representation of push–pull arrows in the figures of our model.
- *Constructive memory* [fourth paragraph in Section 3.1]: This concept is included into our framework in terms of activities 4–6 in Figure 5.
- *First-person knowledge instead of encoded knowledge* [sixth paragraph in Section 3.1]: This concept is shown in our descriptions of the situated FBS representations of processes in Sections 3.2 and 3.3: all representations of the function, behavior, and structure of a process are presented as a function of different views and purposes, which are clearly first-person constructions rather than third-person encodings."

6.3. Question 2

"Your work requires further clarification of situatedness in the model to understand it fully. Could you elaborate?"

6.4. Answer 2

"A clarification of situatedness is provided in our answer above. In addition, we have elaborated the notion of situatedness by including a further section [Section 4] on the situated design of processes."

6.5. Question 3

"Is your model useful and where would it be used? Could you devise an example with a detailed process analysis showing how your ontology framework would be useful and explained in a systemic manner?"

6.6. Answer 3

"We described the motivation for a process ontology, in particular, one that includes higher level semantics [Section 1]. Examples for how our ontology can be used to compare and classify different processes in design were presented [Section 2.2.4]. A detailed example that illustrates in a systemic manner how the FBS ontology can be used to represent the situated design of processes was provided [Section 4.2]."

ACKNOWLEDGMENTS

This research is supported by Australian Research Council Grant DP0559885.

REFERENCES

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

Bickhard, M.H., & Campbell, R.L. (1996). Topologies of learning. *New Ideas in Psychology 14(2)*, 111–156.

Chandler, A.D. (1962). *Strategy and Structure.* Cambridge, MA: MIT Press.

Chandrasekaran, B., Goel, A.K., & Iwasaki, Y. (1993). Functional representation as design rationale. *IEEE Computer 26(1)*, 48–56.

Chandrasekaran, B., & Josephson, J.R. (2000). Function in device representation. *Engineering with Computers 16(3–4)*, 162–177.

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

Clibbon, K., & Edmonds, E. (1996). Representing strategic design knowledge. *Engineering Applications of Artificial Intelligence 9(4)*, 349–357.

Cross, N. (1994). *Engineering Design Methods: Strategies for Product Design.* Chichester: Wiley.

Cummings, S., & Wilson, D., Eds. (2003). *Images of Strategy.* Oxford: Blackwell.

de Kleer, J., & Brown, J.S. (1984). A qualitative physics based on confluences. *Artificial Intelligence 24*, 7–83.

Dewey, J. (1896). The reflex arc concept in psychology. *Psychological Review 3*, 357–370.

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

Gero, J.S. (1999). Constructive memory in design thinking. In *Design Thinking Research Symposium: Design Representation* (Goldschmidt, G., & Porter, W., Eds.), pp. 29–35. Cambridge, MA: MIT.

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

Gero, J.S., & Kannengiesser, U. (2004). The situated function–behaviour–structure framework. *Design Studies 25(4)*, 373–391.

Grabowski, H., Rude, S., & Grein, G., eds. (1998). *Universal Design Theory.* Aachen, Germany: Shaker Verlag.

Gruber, T.R. (1989). Automated knowledge acquisition for strategic knowledge. *Machine Learning 4*, 293–336.

Haymaker, J., & Fischer, M. (2001). *Challenges and Benefits of 4D Modeling on the Walt Disney Concert Hall Project*, CIFE Working Paper 64. Stanford, CA: Center for Integrated Facility Engineering, Stanford University.

Hori, K. (2000). An ontology of strategic knowledge: key concepts and applications, *Knowledge-Based Systems 13*, 369–374.
Hubka, V., & Eder, W.E. (1996). *Design Science: Introduction to the Needs, Scope and Organization of Engineering Design Knowledge*. Berlin: Springer–Verlag.
International Alliance for Interoperability. (2006). *Industry Foundation Classes IFC2x* (3rd ed.). Accessed at http://www.iai-international.org/Model/R2x3_final/index.htm
Kitamura, Y., Kashiwase, M., Fuse, M., & Mizoguchi, R. (2004). Deployment of an ontological framework of functional design knowledge. *Advanced Engineering Informatics 18(2)*, 115–127.
Mintzberg, H., & Waters, J.A. (1985). Of strategies, deliberate and emergent. *Strategic Management Journal 6(3)*, 257–272.
Motus, L., & Rodd, M.G. (1994). *Timing Analysis of Real-Time Software.* Oxford: Pergamon Press.
Murdock, J.W., & Goel, A. (2001). Meta-case-based reasoning: using functional models to adapt case-based agents. *Proc. Int. Conf. Case-Based Reasoning 2001* (Aha, D.W., & Watson, I., Eds.), pp. 407–421. Berlin: Springer.
NIST. (1993). *Integration Definition for Function Modeling (IDEF0)*, Federal Information Processing Standards Publication 183. Gaithersburg, MD: National Institute of Standards and Technology.
NIST. (2000). *The Process Specification Language (PSL): Overview and Version 1.0 Specification*, NIST Internal Report 6459. Gaithersburg, MD: National Institute of Standards and Technology.
NIST. (2004). *Inputs and Outputs in the Process Specification Language*, NIST Internal Report 7152. Gaithersburg, MD: National Institute of Standards and Technology.
Schön, D.A. (1983). *The Reflective Practitioner: How Professionals Think in Action.* New York: Harper Collins.
Schön, D.A., & Wiggins, G. (1992). Kinds of seeing and their functions in designing. *Design Studies 13(2)*, 135–156.
Sim, S.K., & Duffy, A.H.B. (1998). A foundation for machine learning in design. *Artificial Intelligence for Engineering Design, Analysis and Manufacturing 12(2)*, 193–209.
Smith, G.J., & Gero, J.S. (2005). What does an artificial design agent mean by being "situated"? *Design Studies 26(5)*, 535–561.
Stone, R.B., & Wood, K.L. (2000). Development of a functional basis for design. *Journal of Mechanical Design 122(4)*, 359–370.
Stroulia, E., & Goel, A. (1995). Functional representation and reasoning in reflective systems. *Journal of Applied Intelligence 9(1)*, 101–124.
Suwa, M., Gero, J.S., & Purcell, T. (1999). Unexpected discoveries and s-inventions of design requirements: a key to creative designs. In *Computational Models of Creative Design IV* (Gero, J.S., & Maher, M.L., Eds.), pp. 297–320. Sydney, Australia: University of Sydney, Key Centre of Design Computing and Cognition.
von der Weth, R. (1999). Design instinct?—The development of individual strategies. *Design Studies 20(5)*, 453–463.
Wiest, J.D., & Levy, F.K. (1977). *A Management Guide to PERT/CPM.* Englewood Cliffs, NJ: Prentice–Hall.
Ziemke, T. (1999). Rethinking grounding. In *Understanding Representation in the Cognitive Sciences: Does Representation Need Reality?* (Riegler, A., Peschl, M., & von Stein, A., Eds.), pp. 177–190. New York: Plenum Press.

John S. Gero is a Research Professor at the Krasnow Institute for Advanced Study and Volgenau School of Information Technology and Engineering and in the Department of Computer Science, George Mason University, and Visiting Professor at the Massachusetts Institute of Technology. Formerly he was a Professor of Design Science and Co-Director of the Key Centre of Design Computing at the University of Sydney. Dr. Gero is the author/editor of 43 books and has published over 550 research papers. He has been a Visiting Professor of architecture, civil engineering, cognitive psychology, computer science, design and computation, and mechanical engineering in the United States, United Kingdom, France, and Switzerland. His research focuses on computational, cognitive, and neurocognitive studies of designing.

Udo Kannengiesser is a Researcher in the Empirical Software Engineering Program at NICTA, Australian's Centre of Excellence for information and communication technology. He holds a degree in mechanical engineering from the University of Karlsruhe, Germany, with a specialization in information systems for design and production. Dr. Kannengiesser obtained his PhD at the Key Centre of Design Computing and Cognition at the University of Sydney.