A FUNCTION-BEHAVIOUR-STRUCTURE ONTOLOGY OF PROCESSES

JOHN S GERO AND UDO KANNENGIESSER
University of Sydney, Australia

Abstract. This paper shows how the function-behaviour-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 demonstrate that this ontology supports a situated view of processes based on a model of three interacting worlds.

1. Introduction

Ontologies are structured conceptualisations 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 artefacts. They form the basis for a common understanding and terminological agreement on all relevant properties of a specific artefact or class of artefacts. Ontologies can then be used to represent the evolving states of designing these artefacts 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 and Eder (1996)). However, most of this work has not been based on process ontologies, which makes comparison of the different taxonomies difficult. 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 4D CAD systems links 3D object models to project schedules (Haymaker and Fischer 2001). Process ontologies used in the design field include Cyc (Lenat and Guha 1990), IDEF0 (NIST 1993) and PSL (NIST 2000).

Most process ontologies and representations have a view of processes that is based on activities and/or their pre- and post-conditions. Higher-level semantics 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 higher-level semantics is Gero's (1990) function-behaviour-structure (FBS) ontology. Its original focus was on representing artificial objects. In this paper we show how this focus can be extended to include processes. We then apply Gero and Kannengiesser's (2004) three-world model to develop a situated view of processes, which also demonstrates some of the benefits of including higher-level semantics into process representations.

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. *Function* (F) of an object is defined as its teleology, i.e. "what the object is for".
2. *Behaviour* (B) of an object is defined as the attributes that are derived or expected to be derived from its structure (S), i.e. "what the object does".
3. *Structure* (S) of an object is defined as its components and their relationships, i.e. "what the object consists of". The structure (S) of most objects can be described in terms of geometry, topology and material.

Humans construct connections between F, B and S through experience and through the development of causal models based on interactions with the object. Specifically, function (F) is ascribed to behaviour (B) by establishing a teleological connection between the human's goals and observable or measurable effects of the object. Behaviour (B) is causally connected to structure (S), i.e. it can be derived from structure using physical laws or heuristics. There is no direct connection between function (F) and structure (S), which is known as the "no-function-in-structure" principle (De Kleer and 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 (S): Architects typically view this structure as a configuration of spaces, while 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 modelling objects as assemblies composed of sub-assemblies and individual parts. Each of these components can again contain other sub-assemblies 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 visualised and understood. Processes are more abstract constructs that include transitions from one state of affairs to another.

The well-established field of object-oriented software engineering has most explicitly demonstrated how abstraction can overcome the traditional division between the object-centred and the process-centred view of the world. Object-oriented software commonly uses a set of program elements that are conceived of as representing objects as well as processes that operate on them. All of these program elements encapsulate state variables and define methods to enable interactions with other elements.

The high-level categorisations provided by the FBS ontology can be used to create a similar, 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. While 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 (F), behaviour (B) and structure (S).

It is not hard to see that the notion of function (F) 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.

Behaviour (B) 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 behaviours (B) of processes are speed, rate of convergence, cost, amount of space required and accuracy.

While process function (F) and process behaviour (B) are not fundamentally different to object function and object behaviour, process

structure (S) is clearly distinctive. It includes three classes of components and two classes of relationships, Figure 1.

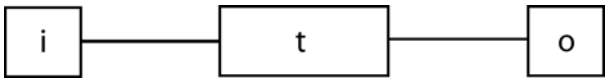

Figure 1. The structure (S) of a process. (i = input; t = transformation; o = output)

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).

2.2.1. Input (i) and Output (o)

The input (i) and the output (o) structure elements represent properties of other entities in terms of their variables and/or their values. For example, the process of transportation changes only the values for the location of a (physical) object (e.g. the values of its x-, y- and z-coordinates). As the input (i) and output (o) contain the same variables here, such a process can be characterised as homogenous. Heterogenous processes, in contrast, use disparate variables as input (i) and output (o). For example, the process of electricity generation takes mechanical motion as input (i) and produces electrical energy as output (o).

Input (i) and output (o) 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 (i) or to return procedure calls as their output (o). All variables and values used as input (i) and output (o) of a process may refer to the function, behaviour or structure of other objects or processes.

2.2.2. Transformation (t)

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 a UML[1] activity diagram. Such descriptions are often viewed as a collection of subordinate processes. In the software example, this is most explicit when a procedure

[1] Unified Modeling Language

calls other procedures that are possibly located in other program components or other computers. Every sub-process can again be modelled in terms of function, behaviour and structure.

The transformation (t) of a process can also be described in terms of an object. Take the software example, the transformation (t) of a process may be viewed simply as the object (in the context of object-oriented programming) that provides appropriate methods to carry out that process. Another example for a transformation (t) can be a computational agent. Such object-centred descriptions of transformations (t) are often used when not much is known about the internal mechanisms of that transformation or when not much is gained by explicitly modelling these mechanisms. In some cases, the transformation (t) is only a "black box" that merely serves to connect the input (i) to the output (o). For example, the programmer designing the software procedure constructs an extensive set of properties related to the transformation (t). In contrast, for the users of that procedure the transformation (t) is often a "black box", as the source code is usually not available or relevant. They base their views of the process structure (S) mainly on the input (i) and output (i) variables that are specified in the application programming interface (API).

2.2.3. Relationships

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

2.2.4. Some 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, behaviour or structure. Take Gero's (1990) eight fundamental classes of processes involved in designing, they can be distinguished by differences in their input (i) and output (o). For example, while synthesis is a transformation of expected behaviour (i) into structure (o), analysis transforms structure (i) into behaviour (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 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), etc. Other process instances can be based on the transformation (t). For example, the synthesis of a design object can be carried out using a range of different transformations (t)

or techniques to map expected behaviour onto structure. Examples include case-based reasoning, genetic algorithms or gradient-based search methods.

While most process classifications and taxonomies are based on differences in structure (S), processes can also be distinguished according to their behaviour (B) and function (F). For example, design optimization processes can be characterised on the basis of differences in their speed, differences in the amount of space they require or other behaviours (B). Another example has been provided by Sim and Duffy (1998), who propose a multi-dimensional classification of machine learning processes in design that can partially be mapped on structure (S) and function (F) of a process. Specifically, learning processes are grouped according to input knowledge (i), knowledge transformers (t), output knowledge (o) and learning goal (F), among others.

3. Situated FBS Representations of Processes

3.1. SITUATEDNESS

Designing is an activity during which designers perform actions in order to change their environment. By observing and interpreting the results of their actions, they then decide on new actions to be executed on the environment. This means that the 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).

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, 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. The relevance of this notion in the area of design research has been shown by Gero (1999). Constructive memory is best exemplified by a quote from Dewey via 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 uses 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. 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.

Gero and Kannengiesser (2004) have modelled situatedness as the interaction of three worlds, each of which can bring about changes in any of

the other worlds. The three worlds include the observer's external world, interpreted world and expected world, Figure 2.

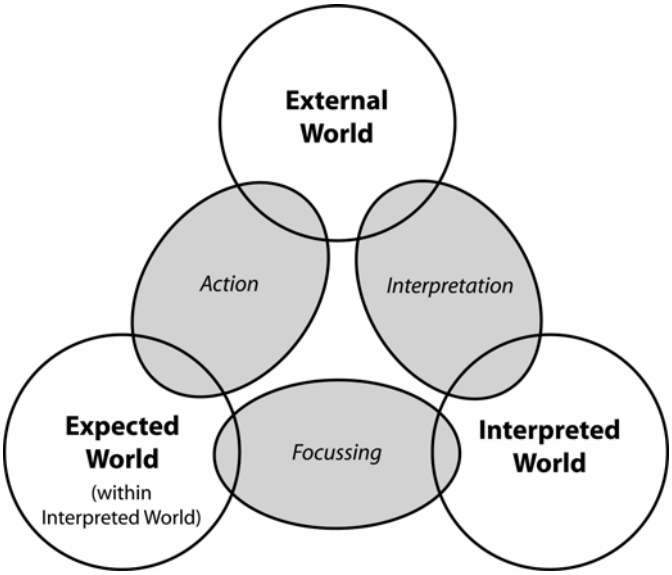

Figure 2. Situatedness as the interaction of three worlds.

The *external world* is the world that is composed of representations outside the observer (or designer).

The *interpreted world* is the world that is built up inside the designer in terms of sensory experiences, percepts and concepts. It is the internal representation of that part of the external world that the designer interacts with.

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

These three worlds are linked together by three classes of connections. *Interpretation* transforms variables which are sensed in the external world into the interpretations of sensory experiences, percepts and concepts that compose the interpreted world. *Focussing* takes some aspects of the interpreted world, uses them as goals in the expected world and suggests actions, which, if executed in the external world should produce states that reach the goals. *Action* is an effect which brings about a change in the external world according to the goals in the expected world.

3.2. CONSTRUCTING DIFFERENT VIEWS FOR DIFFERENT PURPOSES

Gero and Kannengiesser's (2004) three-world model can be used to construct a situated FBS view of processes. The main basis for creating situated representations is the distinction between the external and the interpreted world. Locating function (F), behaviour (B) and structure (S) of a process in each of these worlds, Figure 3, results in six ontological categories:

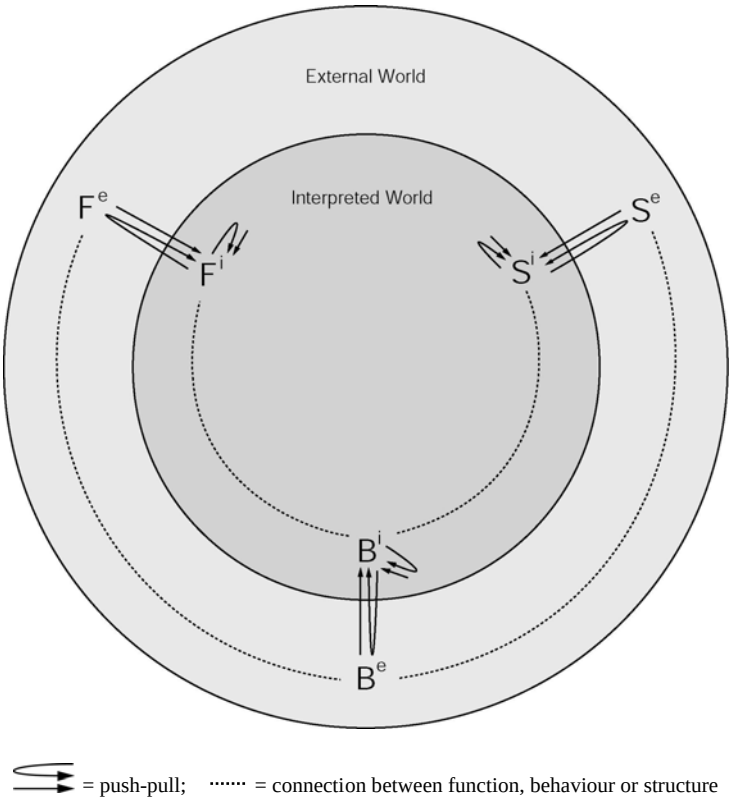

Figure 3. External and interpreted FBS representations of processes.

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

Process representations of categories 4, 5 and 6 are generated via push-pull mechanisms involving only the internal world (constructive memory) or both internal and external worlds (interpretation).

3.2.1. External vs. 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 (S). This is due to a number of reasons.

First, many instances of external process structure (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 entities in the external world then requires acts of discretisation 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 (t) or, alternatively, as part of its output (o).

Second, the kind of components of the process structure (S) 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 (t) from a detailed procedural plan to an object or a simple "black box". There are also many examples for disparate views of the input (i) and output (o) 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 the output of the same process in terms of (material, labour, etc.) cost and yield, respectively. Similar view-dependent examples have been presented by NIST (2004).

3.2.2. External vs. Interpreted Behaviour of a Process

The distinction between external and interpreted worlds is also useful when dealing with the performance or behaviour (B) 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 (available in the external world as external behaviour (B^e)) 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 behaviour (B) is usually worth considering for users only if their hardware resources are limited for current purposes.

The kind of interpreted behaviour (B^i) that an observer is interested in also affects the way in which that observer interprets the structure (S) that is responsible for causing that behaviour. This is the case when no external behaviour (B^e) and no memories of previous interpreted behaviour (B^i) are available, and the interpreted behaviour (B^i) must be derived from structure. If, for instance, the speed of a process is to be measured, then a structural

description of the input (i) and output (o) 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 (t) for given inputs (i) and outputs (o).

3.2.3. External vs. Interpreted Function of a Process

The need to separate the interpreted from the external world is most obvious for the function (F) 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 interpreted process functions (F^i), which may be independent of the constraints imposed by process structure and behaviour. 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 behaviour) may be given.

3.3. CONSTRUCTING DIFFERENT PURPOSES FROM DIFFERENT VIEWS

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

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

The distinction between the interpreted and the expected world reflects the potential gap between the perceived and the desired state of the world. Such a gap usually results in an action to change the external world according to the goals in the expected world.

3.3.1. External, Interpreted and Expected Structure of a Process

Representations of process structure (S) in the expected world describe the composition of desired processes. Actions can then be performed to realise (represent) the desired processes in the external world. One example of such processes is a strategy. One distinguishing feature of strategies is that the transformation (t) components of their structure (S) are viewed as actions or sequences of actions, undertaken either by individuals (Gruber 1989) or by

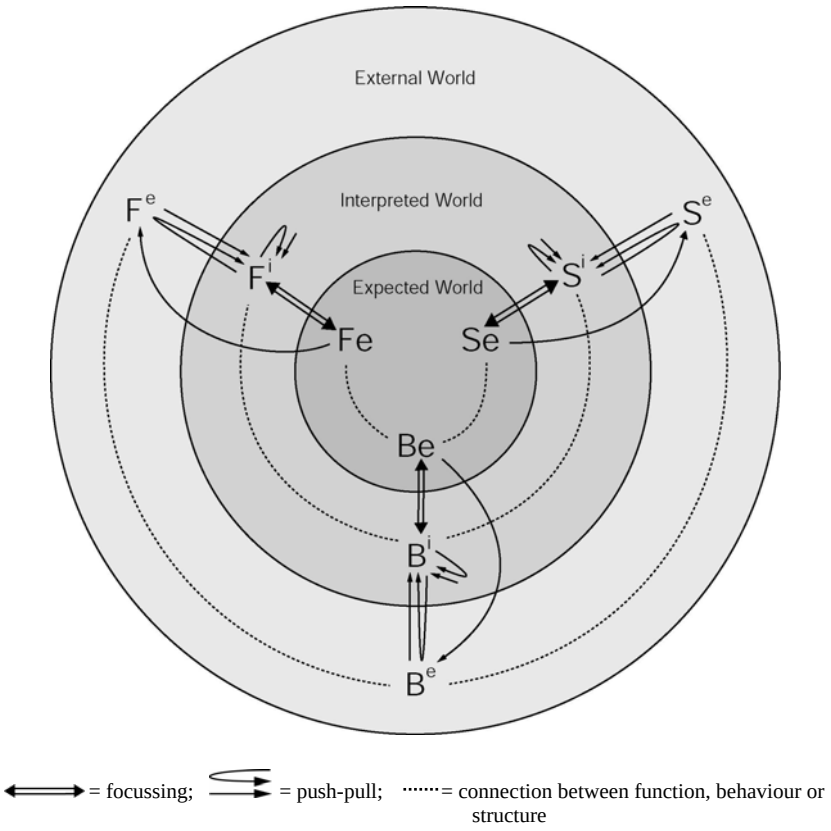

Figure 4. External, interpreted and expected FBS representations of processes.

organisations (Chandler 1962). These actions can then be interpreted again as part of an interpreted process structure (S^i) that may be different from the initial, expected process structure (Se^i). New strategies can be adopted by transferring interpreted process structures (S^i) into the expected world.

The interaction between the external, interpreted and expected structure (S) 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 and Wilson 2003). Strategizing combines the traditional idea of top-down implementation of pre-formed strategies with more recent models of bottom-up recognition of new strategies as "patterns in a stream of actions" (Mintzberg and Waters 1985).

It has frequently been suggested that new strategies are recognised by identifying and eliminating redundant steps (Roberts and Newton 2001). This complies with the notion of emergence, which is a general mechanism for deriving new design concepts (Gero 1996). Emergence of design strategies has been demonstrated by Nath and Gero (2004) to allow a computational system to acquire and reuse search (process) knowledge encoded as rules. The system can identify mappings between past design contexts and design decisions that led to useful results in these contexts. It then constructs new rules from these mappings using explanation-based learning.

Besides emergence, a number of other mechanisms may bring about new strategies. These are mutation, combination, analogy and first principles (Gero 1996). Not much research has been undertaken to date to apply them to strategies.

3.3.2. External, Interpreted and Expected Behaviour of a Process

Differences between the interpreted and the expected world at the level of the behaviour (B) of a process are, for instance, 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 expected behaviour (Be^i) of the process. First, corrective action may be taken to change the current process performance in the external world (B^e) that would then change the corresponding interpreted performance (B^i). Second, the expected behaviour (Be^i) may be adjusted to the current state of the process in order to satisfice the project plan.

The performance or behaviour (B) level has also been used to justify the selection of a particular design strategy (Clibbon and Edmonds 1996). Chandrasekaran et al. (1993) have similarly included behaviour (B) into representations of design rationale to retrospectively document and explain decisions taken in a design process. The distinction between interpreted and expected process behaviour (B) allows comparing the performance of alternative strategies and ultimately selecting one of them.

3.3.3. External, Interpreted and Expected Function of a Process

The distinction between interpreted and expected function (F) of a process describes the gap between potentially adoptable and currently focussed purposes ascribed to the process. Similar to behaviour (B), this gap may be reduced or eliminated through action to modify external function (F^e) or through adoption of new expected function (Fe^i).

Representations of expected function (F) can also be used to provide constraints for selecting the behaviour (B) and structure (S) of processes via the connections between function, behaviour and structure. They link the performance and composition of processes to the current teleological context by adding functional requirements. For example, von der Weth (1999) has suggested that expectations of functions (F) such as “carefulness” or “thoughtfulness” support the selection of strategies that are adapted to the degree of complexity, novelty and dynamism of a given situation.

4. Discussion

We have presented the FBS ontology as a structured conceptualisation of the domain of processes. We claim that any class of process can be represented using this ontology. A number of examples of processes in the design domain have been described earlier in this paper. Our ontology provides a uniform representation that allows locating as well as distinguishing between them.

Integrating function and behaviour in a process ontology adds higher-level 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 select, 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, behavioural and structural levels increases flexibility and applicability of the system in different situations.

Another major advantage of the presented FBS ontology of processes is that it uses the same fundamental constructs – function, behaviour 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, specialised cognitive mechanisms. As everything in the world looks the same when viewed in terms of FBS, only one cognitive mechanism is required. Reflective, meta-cognitive systems (e.g. Singh et al. (2004)) would particularly benefit from our ontological approach to processes as it avoids implementing multiple layers of reasoning.

Acknowledgements

This research is supported by a grant from the Australian Research Council.

References

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

Bickhard, MH and Campbell, RL: 1996, Topologies of learning, *New Ideas in Psychology* **14**(2): 111-156.

Chandler, AD: 1962, *Strategy and Structure*, MIT Press, Cambridge.

Chandrasekaran, B, Goel, AK and Iwasaki, Y: 1993, Functional representation as design rationale, *IEEE Computer* **26**(1): 48-56.

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

Clibbon, K and 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*, John Wiley & Sons, Chichester.

Cummings, S and Wilson, D (eds): 2003, *Images of Strategy*, Blackwell Publishers, Oxford.

De Kleer, J and Brown, JS: 1984, A qualitative physics based on confluences, *Artificial Intelligence* **24**: 7-83.

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

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

Gero, JS: 1999, Constructive memory in design thinking, *in* G Goldschmidt and W Porter (eds), *Design Thinking Research Symposium: Design Representation*, MIT, Cambridge, MA, pp. 29-35.

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.

Gruber, TR: 1989, Automated knowledge acquisition for strategic knowledge, *Machine Learning* **4**: 293-336.

Haymaker, J and Fischer, M: 2001, Challenges and benefits of 4D modeling on the Walt Disney concert hall project, *CIFE Working Paper #64*, Center for Integrated Facility Engineering, Stanford University, Stanford, CA.

Hori, K: 2000, An ontology of strategic knowledge: Key concepts and applications, *Knowledge-Based Systems* **13**: 369-374.

Hubka, V and Eder, WE: 1996, *Design Science: Introduction to the Needs, Scope and Organization of Engineering Design Knowledge*, Springer-Verlag, Berlin.

Lenat, DB and Guha, RV: 1990, *Building Large Knowledge-Based Systems: Representation and Inference in the Cyc Project*, Addison-Wesley, Reading.

Mintzberg, H and Waters, JA: 1985, Of strategies, deliberate and emergent, *Strategic Management Journal* **6**(3): 257-272.

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

NIST: 1993, Integration definition for function modeling (IDEF0), *Federal Information Processing Standards Publication 183*, National Institute of Standards and Technology, Gaithersburg, MD.

NIST: 2000, The process specification language (PSL): Overview and version 1.0 specification, *NIST Internal Report 6459*, National Institute of Standards and Technology, Gaithersburg, MD.

NIST: 2004, Inputs and outputs in the process specification language, *NIST Internal Report 7152*, National Institute of Standards and Technology, Gaithersburg, MD.

Roberts, MJ and Newton, EJ: 2001, Understanding strategy selection, *International Journal of Human-Computer Studies* **54**: 137-154.

Schön, DA: 1983, *The Reflective Practitioner*, Harper Collins, New York.

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

Sim, SK and Duffy, AHB: 1998, A foundation for machine learning in design, *Artificial Intelligence for Engineering Design, Analysis and Manufacturing* **12**(2): 193-209.

Singh, P, Minsky, M and Eslick, I: 2004, Computing commonsense, *BT Technology Journal* **22**(4): 201-210.

Smith, GJ and Gero, JS: 2005, What does an artificial design agent mean by being 'situated'?, *Design Studies* **26**(5): 535-561.

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.

von der Weth, R: 1999, Design instinct? – The development of individual strategies, *Design Studies* **20**(5): 453-463.

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.