A formalization of Ashok Goel's SBF concept of function

Stefano Borgo[1], Massimiliano Carrara[2], Pawel Garbacz[3], and Pieter E. Vermaas[4]

[1] Laboratory of Applied Ontology, ISTC CNR, Trento, Italy
stefano.borgo@cnr.it
[2] FISPPA Department – Section of Philosophy, University of Padua, Italy
massimiliano.carrara@unipd.it
[3] Philosophy Department, John Paul II Catholic University of Lublin, Poland
garbacz@kul.lublin.pl
[4] Philosophy Department, Delft University of Technology, The Netherlands
p.e.vermaas@tudelft.nl

Abstract. We formalize within the DOLCE foundational ontology the Structure-Behavior-Function model (SBF) proposed by Ashok K. Goel and colleagues. Our work focuses in particular on the notion of function. This work on SBF is part of a larger project that includes the formalization of the concepts of function by Chandrasekaran and Josephson and by Stone and Wood. The overall goal is to make engineering functional descriptions of technical artifacts based on different concepts of function, exchangeable by separately formalizing these different concepts in a single ontological framework. The formalization is a necessary step towards the development of an integrated information system for engineering design.

Keywords: function, formal ontology, SBF model, Ashok K. Goel, DOLCE.

Introduction

The aim of this contribution is to formalize the concept of function of technical artifacts as advanced by Ashok K. Goel and his colleagues [10] as part of the *Structure-Behavior-Function* (SBF) model. The SBF concept of function is developed in [10, pp. 25-26], [11] from the so-called *Functional Representation* (FR) approach towards modeling functions, proposed by Chandrasekaran and Josephson [8]. The SBF model extends this original modeling, for instance, by describing the structure of technical artifacts in terms of components and substances, and by adding the assumption that there exists a limited set of primitive functions.

Given the relationship between the SBF model and the FR approach we arrive at a formalization of the SBF concept of function using as a starting point our earlier formalization of the FR approach. The formalization of SBF functions includes also formal characterizations of the SBF concepts of structure and behavior: we take a behavior in SBF to be a discrete sequence of states, and an

SBF function to be an SBF behavior with a fixed input state and a fixed output state, both specified by a set of values for state parameters (pre-conditions and post-conditions). An SBF function is then formalized as constraints on these parameters of states.

A central starting point in our larger project is to formalize all engineering concepts of function within the same ontology, seen as a unifying structure for the analysis and the formalization of these concepts, namely the *Descriptive Ontology for Linguistic and Cognitive Engineering* (DOLCE) [12].

The paper opens in section 1 with a brief description of our larger project. Section 2 outlines the central concepts of DOLCE. Then, in Section 3, we describe the SBF model in some detail and relate it with the FR approach. In section 4 we focus on the formalization of the SBF concept of function.

1 The larger project

This work on SBF functions is part of a larger project aimed at making engineering functional descriptions of technical artifacts (based on different concepts of function) interoperable. This is obtained by separately formalizing the main different concepts within a single ontological framework. The approach is described and argued for in [5]:

> [It] does not aim directly at a single concept of function, but tries to reconstruct the main meanings that engineers attach to this term by means of a series of formalizations within one single formal framework. In this strategy one focuses still on well-defined and specific concepts of function, which are taken as classical concepts, but now different [meanings] of such concepts are formalized. [It] is also in conformance with engineering practice by describing and formalizing the concepts of function used. [...] Yet this [...] strategy disambiguates functional descriptions only in a weak sense. Each meaning that is formalized on this strategy is analyzed in detail, assessed for consistency, and if needed at points corrected. And if such corrections are not feasible, particular meanings may even be discarded as untenable ones [...]. Yet, after formalization it still amounts to different concepts of function that co-exist in one formal system. By their co-existence in one formal system, these functional concepts may be compared and related, just as any other set of concepts can be compared and related. [...] [5, p. 152]

In our larger project we thus accept the co-existence of different meanings of function as a feature of engineering [15], and proceed by formalizing those different meanings. In [1] we formalized the concept of function by Chandrasekaran and Josephson [8], which represents the FR approach towards modeling functions. In [2] we formalized the Stone and Wood [14] concept of functions, representing the FB modeling approach. And in [9] we provided a formal comparison between these two formalizations and showed how automatic exchange of functional descriptions originating in these approaches may look like. With this contribution we proceed in our project by including a formalization of Goel's SBF concept of function.

2 A very brief introduction to DOLCE

2.1 The general structure of DOLCE

Dolce is a foundational ontology of particulars with a clear cognitive bias since its categories are obtained by analyzing the surface structure of language and cognition. Consequences of this approach are that DOLCE's categories are at the so-called *mesoscopic level*, the level of the middle-sized objects we, as humans, perceive. The Dolce's taxonomic structure is pictured in Figure 1. Each node in the graph is a category of the ontology. A category that is a direct subcategory of another is depicted by drawing the latter higher in the graph and linking them with an edge. Particular is the top category. The set of direct subcategories of a given category forms a partition unless dots are inserted.

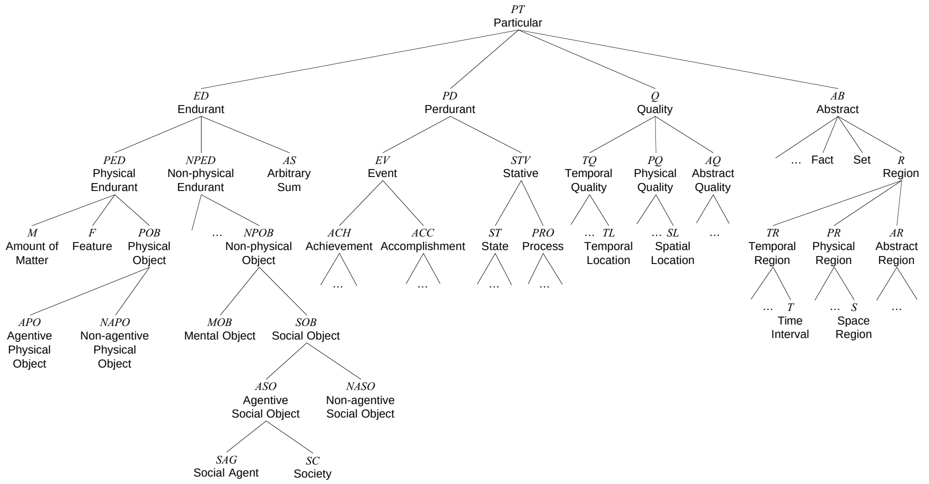

Fig. 1. The DOLCE taxonomy (from [12]).

The DOLCE ontology category ENDURANT comprises objects, e.g., a *hammer*, and amounts of matter, e.g., *the amount of water in this glass, the amount of gold in my wedding ring*, while the category PERDURANT comprises events like *making a hole* or *a soccer game*, that is, things that happen in time. The term 'object' is used in the ontology to capture a notion of unity as suggested by the partition of the class PHYSICAL ENDURANT into classes AMOUNT OF MATTER, FEATURE, and PHYSICAL OBJECTS (see Figure 1). Among those we need to explain in more detail the DOLCE notion of FEATURE. In DOLCE, features are dependent entities which are wholes, thus distinguished from individual qualities:

> Typical examples of features are "parasitic entities" such as holes, boundaries, surfaces, or stains, which are generically constantly dependent on physical objects (their hosts). All features are essential wholes, but, as in the case of objects, no common unity criterion may exist for all of them. However, typical features have a topological unity, as they are singular entities. Some features may be *relevant parts* of their host, like a bump or an edge, or *places* like a hole in a piece of cheese, the underneath of a table, the front of a house, which are not parts of their host. [12, p. 16]

2.2 DOLCE categories and relations we focus on

In this section we present the categories of DOLCE in Figure 1 that are relevant to our work. Note that the terminology adopted departs sometimes from that in engineering design, knowledge representation, and conceptual modeling since affected in part by the philosophical literature.

$\mathsf{ED}(\mathtt{x})$ stands for "$\mathtt{x}$ is an endurant". An endurant is an entity that is wholly present at any time it is present. It is physical if located in space and time: a *hammer_#321*, a *mover machine_#111*, an *amount of plastic*, and *the cavity in which a piston moves*.

$\mathsf{PED}(\mathtt{x})$, a subcategory of ED, stands for "$\mathtt{x}$ is a physical endurant. A *hammer*, a *mover machine*, an *amount of plastic*, and *the cavity in which a piston moves*, are all examples of physical endurants. We will use two subcategories of physical endurants: physical objects POB and features F.

$\mathsf{NPED}(\mathtt{x})$ stands for "$\mathtt{x}$ is a non-physical endurant." NPED is a subcategory of ED that includes mental objects, e.g., beliefs, intentions, etc., and social objects (SOB), e.g., norms, shares, peace treaties.

$\mathsf{PD}(\mathtt{x})$ stands for "$\mathtt{x}$ is a perdurant", i.e., an entity that is only partially present at any time that is present. For instance, consider the perdurant *producing an item of type #234* that consists of riveting two metal pieces and painting the resulting piece. While the *painting* goes on, the (temporal) part corresponding to *riveting* is no longer present and when this is present, the *painting* still has to come. We will use also the basic distinction between events (EV) and states (ST) among perdurants. A perdurant is stative or eventive according to whether it holds of the mereological sum of two of its instances, i.e., if it is cumulative or not. A sitting is a state since the sum of two sittings is still a sitting, while a sitting down is an event since the sum of two sitting downs is not a sitting down.

Among the ontological relations in DOLCE we will make use of the *parthood* relation: "$\mathtt{x}$ is part of $\mathtt{y}$", written $\mathsf{P}(\mathtt{x},\mathtt{y})$. The formal theory based on parthood is called *mereology* [13]. In DOLCE the *parthood* relation applies to pairs of endurants and to pairs of perdurants. For instance, if $\mathtt{a}$ = 'writing article A' and $\mathtt{b}$ = 'writing the introduction to article A', then $\mathsf{P}(\mathtt{b},\mathtt{a})$ holds. For endurants, the relation of parthood is temporalized since an endurant may loose and gain parts throughout its existence: $\mathsf{P}(\mathtt{e},\mathtt{e}',\mathtt{t})$ says that the endurant $\mathtt{e}$ is part of the endurant $\mathtt{e}'$ at the instant or interval $\mathtt{t}$. In the setting of SBF holds the simplifying assumption that the time interval is fixed: consequently, the temporal relativization of mereological parthood between endurants is here neglected.

A number of auxiliary definitions, like proper part, overlap and sum, can be introduced from P. (Symbol $\triangleq$ indicates a definition.)

$$\mathsf{PP}(\mathtt{x},\mathtt{y}) \triangleq \mathsf{P}(\mathtt{x},\mathtt{y}) \wedge \neg\mathsf{P}(\mathtt{y},\mathtt{x}) \tag{1}$$

A perdurant is a *proper part* (PP) of another if it is part of the second and not *vice versa*. *Example*: Reading this section is a proper part of reading the paper.

$$\mathsf{O}(\mathtt{x},\mathtt{y}) \triangleq \exists\mathtt{z}(\mathsf{P}(\mathtt{z},\mathtt{x}) \wedge \mathsf{P}(\mathtt{z},\mathtt{y})) \tag{2}$$

Two perdurants *overlap* (O) if a perdurant exists which is simultaneously part of both. *Example*: 'My drinking on the couch' and 'my watching TV on the couch' have 'my sitting on the couch' as part of both. Regarding mereological sum (+), a perdurant $\mathtt{z}$ is the sum of $\mathtt{x}$ and $\mathtt{y}$ provided that $\mathtt{x}$, $\mathtt{y}$ are parts of $\mathtt{z}$, and that whatever overlaps $\mathtt{z}$ also overlaps $\mathtt{x}$ or $\mathtt{y}$. Formally,

$$\mathtt{x}+\mathtt{y} \triangleq \iota\mathtt{z}\,\forall\mathtt{w}(\mathsf{O}(\mathtt{w},\mathtt{z}) \leftrightarrow (\mathsf{O}(\mathtt{w},\mathtt{x}) \vee \mathsf{O}(\mathtt{w},\mathtt{y}))) \tag{3}$$

This definition can be easily extended for ternary, quaternary, etc., operations.

3 Functions in the SBF model and in the FR approach

Here we report the terminology from [10] and connect the concepts used in the SBF model and the concepts advanced in the FR approach. In Section 4, when we formalize SBF concepts, we add more details to the description of these concepts.

An SBF model of an artifact includes submodels of the artifact's structure, behavior and function. These submodels are characterized as follows:

The structural submodel of an artifact consists of a description of the *elements* of the artifact and the *connections* between these elements. In these structural models a distinction is made between elements that are *components* and elements that are *substances*. The connections between components are called *connecting points*.

The behavioral submodel captures the behavior of an artifact in terms of *transitions* between *states* of the artifact, where these states refer to properties of the connecting points of the artifact, that is, of the artifact's structure. The behavioral submodel moreover gives *causal explanations* of these transitions.

Finally, SBF functions describe the role an element in an artifact plays in the operation of the artifacts; an SBF function gives a *purpose* of the element and refers to a behavior by which the element realises the purpose. Some *primitive functions* are listed, e.g., 'create', 'destroy', 'expel', 'allow', 'pump' and 'move'.

Let us now bring in the FR approach as described in [8]. In this approach the term behavior is undestood to have five engineering meanings and the term function to have two. The meanings of behavior are characterized with the help of the primitive notion of *state variable* (the examples are from [8]):

1. the value of some state variable of the artifact or a relation between such values at a particular instant.
2. the value of a property of the artifact or a relation between such values.
3. the value of some state variable of the artifact over an interval of time.
4. the value of some output state variable of the artifact at a particular instant or over an interval.
5. the values of all the described state variables of the artifact at a particular instant or over an interval.

Note that for all meanings, a behavior of a technical artifact is in part *objective* and in part *subjective*. Objective because it eventually depends on the properties or features of the artifact. Still, the very same behavior depends on the designer(s) and, indirectly, on engineering practice for the choice of the variables.

The two meanings of function in the FR approach are called *device-centric* and *environment-centric* meanings. A *device-centric function* of an artifact is a behavior of the artifact that is selected and intended by some agent. The function is described in terms of the properties and behaviors of the artifact only; an example is “making sound” in the case of an electric buzzer. An *environment-centric function* is in turn an effect or impact of this behavior of the artifact on its environment provided this effect or impact is selected and intended by some agent. This kind of function is conceptually separated from the artifact that performs or is expected to perform this function; “enabling a visitor to a house to inform the person inside the house that someone is at the door” is an environment-centric function of the buzzer.

When comparing the concepts advanced in the SBF model and the FR approach, it can be noted that the notions of *behavior* are fairly similar. Moreover, *functions* are derived notions in both: functions give the agent’s viewpoint on behaviors although agents are only implicit in the SBF framework.

In a nutshell: in SBF and in FR *functions* provide the purpose of an entity in a given situation while the entity’s *behavior* is the way the purpose is accomplished. The distinction *device-centric* and *environment-centric* functions is not part of SBF. Here, we will consider the SBF concept of function as typically an FR device-centric function, since – as we will see – SBF functions refer to SBF behaviors and purposes of components that are typically described in terms of properties of the artifact itself, a specification given in FR to device-centric functions.

As concerns *behavior*: in FR the behavior of a technical artifact is the specific way in which the artifact occurs in an event, it is specified by the meanings (1-5) given above, and characterized using the primitive notion of *state variable*; in SBF behavior is also conceived as a specific way in which a technical artifact occurs in an event. Differently from FR, in SBF there is an emphasis on the state-transition construction of behaviors.

Finally, the notion of *structure* is in SBF somewhat more complex than in FR since there is in SBF, and not in FR, a basic distinction between the elements of a device and the connections between the elements.

4 Formalizing SBF Functions

We now develop the formalization of the SBF model starting from our previous work on the FR approach [1], and then extend it to cover the SBF system including the notion of function. The notion of technical artifact (or device) is introduced in SBF without a specific characterization as it happens in FR and the notion of behavior is developed from similar assumptions. Note, however, that the different setting of SBF will later lead us to make some alternative formalization choices.

We identified the following main categories of SBF

- (technical) device and its physical components
- substances
- connections and connection points
- devices' states and behaviors
- functions

Following the methodology described in [5] we first align these categories to the DOLCE taxonomy.

4.1 Ontological categorization

SBF uses a notion of device which is richer than that exploited by FR. SBF can describe to some extent the structure of the device itself. In particular, a basic distinction is set between elements (parts) of the device and connections among them. Elements are clearly divided in: a) physical components, i.e., the physical parts of a device, and b) substances, like fluids and forces. Ontologically these entities are DOLCE's endurants ($\oplus$ stands for the exclusive disjunction):

$$\texttt{Elem}(x) \rightarrow \texttt{PhComp}(x) \oplus \texttt{Subst}(x) \tag{4}$$

$$\texttt{Elem}(x) \rightarrow \mathsf{ED}(x) \tag{5}$$

More specifically, a physical component is a rigid or semi-rigid material object of a subclass (called $\texttt{RigidPOB}$) of physical objects (POB). We do not attempt to constrain this class here since the distinction is not clarified by the authors and does not play a role in the system. A substance can be characterized as an amount of matter (M) or a non-physical endurant (NPED), although not a NPOB, i.e., it is neither a mental nor a social object.

$$\texttt{PhComp}(x) \rightarrow \texttt{RigidPOB}(x) \tag{6}$$

$$\texttt{RigidPOB}(x) \rightarrow \mathsf{POB}(x) \tag{7}$$

$$\texttt{Subst}(x) \rightarrow \mathsf{M}(x) \vee [\mathsf{NPED}(x) \wedge \neg \mathsf{NPOB}(x)] \tag{8}$$

Components, and not substances, may have connection points ($\texttt{ConnPt}$) with which to be connected to other components. In DOLCE these connection points are classified as features (F):

$$\texttt{ConnPt}(x) \rightarrow \mathsf{F}(x) \tag{9}$$

Two connection points in two components can be connected. There is a fixed number of possible connection types depending on how force can be transferred across the connection points: parallel, series, touching, adjoining, bolted, fused, hinged, jointed, tied, telescoped, threaded, frictionally embedded, sewn, nailed, clipped, ball&socket installed and glued. Since connections are relationships needed to discuss force transfer or lack of it, in DOLCE we look at their temporal behavior and classify them in the category of states (ST). Thus by stating that there is a connection of type X between two points we mean that their two components are in a state to exchange force in as much as allowed by the type X of the connection. Classifying connections as states we implicitly add a temporal parameter to the connections. However, as anticipated, we do not exploit temporal information in this formalization.

To capture this, we introduce a ternary relation $\mathtt{Connect}(x, y, z)$ whose intended reading is "connection x holds between connection points y and z (in this order)."

$$\mathtt{Connect}(x, y, z) \rightarrow \mathsf{ST}(x) \wedge \mathtt{ConnPt}(y) \wedge \mathtt{ConnPt}(z) \tag{10}$$

It goes without saying that connections relate different connection points:

$$\mathtt{Connect}(x, y, z) \rightarrow y \neq z \tag{11}$$

As said, behaviors in FR and SBF are similar but the state-transition construction in SBF leads to a somewhat different formalization of behavior, in particular to include causes or explanations for the transitions, an important aspect of SBF. Starting from the notion of behaviour in FR, in the formalization of SBF we add a notion of *system behavior* ($\mathtt{SysBeh}$), namely a perdurant which is a non-empty sequence of states describing at least a connection and at least one transition. We classify transitions as achievements or accomplishments, i.e., in the eventive category EV, see Figure 1. (An interesting alternative would be to model transition types as simplified descriptions of events, this choice would amount to introduce transitions as black box entities.) We use relations $\mathtt{BehStart}$ and $\mathtt{BehEnd}$ to indicate the initial and final states of a transition, respectively, i.e., "$\mathtt{BehStart}$(x,y)" ("$\mathtt{BehEnd}$(x,y)") means that x is the initial (final) state of y.

$$\mathtt{SysBeh}(x) \rightarrow \mathtt{Transition}(x) \tag{12}$$

$$\mathtt{Transition}(x) \rightarrow \mathsf{EV}(x) \tag{13}$$

$$\mathtt{BehStart}(x, y) \vee \mathtt{BehEnd}(x, y) \rightarrow \mathsf{ST}(x) \wedge \mathtt{Transition}(y) \tag{14}$$

We are now ready to discuss functions in SBF. Functions are embedded in the SBF language via a precise list of primitives inspired by the work of Bylander [4]: *create*, *destroy*, *expel*, *allow*, *pump* and *move*. While functions are taken as intended input-output relationships, resembling once again the FR approach, there is an explicit commitment to interpret the behaviors from these elements.

To capture the specific role of these primitives, we add the following axioms (where $\mathtt{Func}(x)$ stays for "x is an SBF function"):

$$[Create(x) \vee Destroy(x) \vee Expel(x) \vee Allow(x) \vee Pump(x) \vee Move(x)] \\ \rightarrow \mathtt{Func}(x) \tag{15}$$

However, these functions are not taken as exhaustive in the SBF language, not even in the sense that any other function should or could be seen as a specialization or a combination of these. Indeed, SBF allows the user to add new functions without restrictions. A basic separation in functional types is given by the mandatory classification of function in achievement (*Achieve*), maintenance (*Maintain*), prevention (*Prevent*) and negation (*Negate*).

$$\texttt{Func}(x) \rightarrow [Achieve(x) \oplus Maintain(x) \oplus Prevent(x) \oplus Negate(x)] \tag{16}$$

From the SBF's examples, these special cases and functions can be classified as social objects in the terminology of DOLCE:

$$\texttt{Func}(x) \rightarrow \mathsf{SOB}(x) \tag{17}$$

However, differently from FR the SBF system makes no direct reference to agents.

4.2 Ontological description

In this section we provide a more detailed ontological characterization of SBF in terms of the four relationships that relate:

1. physical components with physical components: `PhCompOf`
2. physical components with connection points: `HasConnPt`
3. physical components with functions: `HasFunc`
4. functions with behaviors: `FBCorr`

We add relation $\texttt{PhCompOf}(x, y)$, stating that x is a component of (device or component) y, to make explicit the components' structure. We also enforce the existence of a maximal component, namely, the device itself (axiom (20) makes explicit that the SBF models are contextualized to the chosen device). Then, we enforce each component to refer to only one larger component so that the component hierarchy is a tree as requested by SBF:

$$\texttt{PhCompOf}(x, y) \rightarrow \mathsf{PP}(x, y) \tag{18}$$

$$\texttt{PhCompOf}(x, y) \rightarrow \texttt{PhComp}(x) \wedge \texttt{PhComp}(y) \tag{19}$$

$$\exists x \forall y \neg \texttt{PhCompOf}(x, y) \tag{20}$$

$$\texttt{PhCompOf}(x, y_1) \wedge \texttt{PhCompOf}(x, y_2) \rightarrow y_1 = y_2 \tag{21}$$

There is no real difference between components and devices in SBF, thus we do not introduce a specific predicate for devices. The distinction is a matter of focus: components are seen as (functional) parts of larger devices. A component is itself a device from the perspective of any of its subcomponents. Since SBF always concentrates on a single device, any other element in the modeling is a component and components can be nested.

Since the notion of connection point (`ConnPt`) involves the relation of having a connection point, and $\texttt{HasConnPt}(x, y)$ means that x has y as a connection point, we can define the former in terms of the latter:

$$\texttt{ConnPt}(x) \triangleq \exists y \ \texttt{HasConnPt}(y, x) \tag{22}$$

In turn, it seems that $\texttt{HasConnPt}(x, y)$ is ontologically subsumed by the relation of parthood:

$$\texttt{HasConnPt}(x, y) \to \mathsf{PP}(y, x) \tag{23}$$

Note that definition 22 and axioms (6), (7), (9) imply, in DOLCE system, that $\texttt{HasConnPt}(x, y) \to \neg\texttt{PhComp}(y)$. Since components, and not substances, may have connection points, we need:

$$\texttt{HasConnPt}(x, y) \to \texttt{PhComp}(x) \tag{24}$$

Recall that connection points are features, axiom (9), and that substances are material (M) or non-physical endurants (NPED), axiom (8), thus it follows from DOLCE that connection points and substances are distinct.

To relate physical components with their functions we introduce $\texttt{HasFunc}(x, y)$ to mean that component x has function y.

$$\texttt{HasFunc}(x, y) \to \texttt{PhComp}(x) \land \texttt{Func}(y) \tag{25}$$

$$\texttt{Func}(x) \to \exists y\ \texttt{HasFunc}(y, x) \tag{26}$$

$$\texttt{PhComp}(x) \to \exists y\ \texttt{HasFunc}(x, y) \tag{27}$$

As said above, in both SBF and FR functions are derived notions: they select a "reading" of behaviors, and so (perhaps implicitly) provide the agent's viewpoint. The reading is given by selecting the purpose of an entity in a given situation and by considering the entity's behavior as the way that purpose is accomplished. We already stated in (25) that each component in SBF has a function. We can now state a correspondence ($\texttt{FBCorr}$) between functions and behaviors:

$$\texttt{FBCorr}(x, y) \to \texttt{Func}(x) \land \texttt{SysBeh}(y) \tag{28}$$

$$\texttt{Func}(x) \to \exists y\ \texttt{FBCorr}(x, y) \tag{29}$$

$$\texttt{FBCorr}(x, y_1) \land \texttt{FBCorr}(x, y_2) \to y_1 = y_2 \tag{30}$$

Finally, we provide a further characterization of the SBF notion of behavior. Axiom (31) states that a system behavior is the event sum of the states of 'behavior start' and 'behavior end' *plus* the transition event between them (the sum is ordered since they have a temporal dimension). Axiom (32) states that these system behaviors are uniquely identified by their input and output states.

$$\begin{aligned}&\texttt{SysBeh}(x) \to \\ &\quad \exists y, v, z\ [\texttt{BehStart}(y, x) \land \texttt{BehEnd}(v, x) \land \texttt{Transition}(z) \land x = y + z + v]\end{aligned} \tag{31}$$

$$\begin{aligned}&(\texttt{BehStart}(x, z_1) \land \texttt{BehEnd}(y, z_1)) \land (\texttt{BehStart}(x, z_2) \land \texttt{BehEnd}(y, z_2)) \to \\ &\qquad z_1 = z_2\end{aligned} \tag{32}$$

Note that the transition (an event in DOLCE) is naturally directed from the initial state to the ending state and provides the information on how the state change happens, that is, it also includes the causal explanation(s) requested by SBF.

Further formal characteristics. Our ontological characterization of SBF has modeled the explicit ontological aspects of SBF. Below we characterize some key SBF notions in more detail, but this is rather an extension than an explication.

Since PhCompOf is subsumed by the relation of parthood, following [6] we assume that it is a (strict) partial order:

$$\neg\texttt{PhCompOf}(x, x) \tag{33}$$

$$\texttt{PhCompOf}(x, y) \wedge \texttt{PhCompOf}(y, z) \rightarrow \texttt{PhCompOf}(x, z) \tag{34}$$

Furthermore, while the system seems to be extensional, it is unclear whether the mereological reconstruction of these notions requires more specific principles like, e.g., strong supplementation [13, p. 29].

We know that PhComp and HasConnPt are related via axiom (24), but there seem to be an implicit relationship among them stating that each physical component has at least one connection point:

$$\texttt{PhComp}(x) \rightarrow \exists y\ \texttt{HasConnPt}(x, y) \tag{35}$$

Together, they amount to the following equivalence which is easily justified within the engineering perspective:

$$\texttt{PhComp}(x) \leftrightarrow \exists y\ \texttt{HasConnPt}(x, y) \tag{36}$$

Another implicit assumption seem to bind connection points to unique bearers:

$$\texttt{HasConnPt}(x_1, y) \wedge \texttt{HasConnPt}(x_2, y) \rightarrow x_1 = x_2 \tag{37}$$

5 Conclusions

We have studied the concepts underlying Goel's SBF model and proposed a formalization of the system within the DOLCE foundational ontology. The formal characterization of the SBF concepts aimed to cover three key elements: structure, behavior and function. The analysis and the subsequent formalisation show that notions like *component*, *substance* and *connection point*, are only partially characterized and that further information should be collected from other sources, for instance by directly analyzing SBF software packages on component and functional information. We have not investigated this type of material here.

While there are strong connections between the SBF and FR models of function, our analysis shows some important differences which have not been highlight in the literature. The notion of function in SBF does not admit a direct dependence on agents as in FR and, while remaining compatible with the latter, seems to carefully introduce a framework where agents have no explicit role. Furthermore, SBF introduces a short list of functions, showing that function classification is relevant for the framework, but does not include the general distinction between device-centric and environment-centric functions which is at the core of the FR model. Finally, SBF provides the tool for a mereological description of the structure of devices by introducing components and connection ports, while FR focuses mainly on the relations between the devices and their environment.

With this analysis and formalization we are now in the position to formally compare the SBF concept of function with other engineering concepts of function and to extend the means for interoperability across engineering functional descriptions of technical artifacts based on different concepts of function. This will be a subject of future research.

References

1. Borgo, S., Carrara, M., Garbacz, P., Vermaas, P. E., (2009), "A Formal Ontological Perspective on the Behaviors and Functions of Technical Artifacts", Artificial Intelligence for Engineering Design, Analysis and Manufacturing, 23 (1), 3-21.
2. Borgo, S., Carrara, M., Garbacz, P., Vermaas, P. E., (2011), "A Formalization of Functions as Operations on Flows", Journal of Computing and Information Science in Engineering, 11, 031007-031020.
3. Borgo, S., and Leitão, P., (2007), Foundations for a Core Ontology of Manufacturing, in Ontologies: A Handbook of Principles, Concepts and Applications in Information Systems, Integrated Series in Information Systems Vol. 14, ed. by Kishore R., Ramesh R., Sharman R., Springer, New York, pp. 751-776.
4. Bylander, T., (1991), "A Theory of Consolidation for Reasoning about Devices", Man-Machine Studies, 35, 467-489.
5. Carrara, M., Garbacz, P., Vermaas, P. E., (2011), "If Engineering Function is a Family Resemblance Concept: Assessing Three Formalization Strategies", Applied Ontology, 6 (2), 141-163.
6. Casati, R., Varzi, A. C., (2003), Parts and Places: The Structures of Spatial Representation, MIT Press, Cambridge, MA.
7. Chandrasekaran, B., (2005), "Representing Function: Relating Functional Representation and Functional Modeling Research Streams", Artificial Intelligence for Engineering Design, Analysis and Manufacturing, 19 (2), 65-74.
8. Chandrasekaran, B., Josephson, J. R., (2000), "Function in Device Representation", Engineering with Computers, 16 (3/4), 162-177.
9. Garbacz, Borgo, S., Carrara, M., Vermaas, P. E., (2011), "Two Ontology-Driven Formalisations of Functions and Their Comparison", Journal of Engineering Design, 22, 733-764.
10. Goel, A. K., Rugaber, S., Vattam, S., (2009), "Structure, Behavior, and Function of Complex Systems: The Structure, Behavior, and Function Modeling Language", Art. Intelligence for Engineering Design, Analysis and Manufacturing, 23 (1), 23-35.
11. Goel, A. K., (2013), "One Thirty Year Long Case Study; Fifteen Principles: Implications of an AI Methodology for Functional Modeling", Artificial Intelligence for Engineering Design, Analysis and Manufacturing, 27 (3), 203-215.
12. Masolo, C., Borgo, S., Gangemi, A., Guarino, N., Oltramari, A., Schneider, L., (2002), WonderWeb Deliverable D18. Ontology Library (final), WonderWeb European Project, 2003. http://wonderweb.man.ac.uk/deliverables/documents/D18.pdf
13. Simons, P., (1987), Parts: A Study in Ontology, Oxford University Press, Oxford.
14. Stone, R., Wood, K., (2000), "Development of a Functional Basis for Design", Journal of Mechanical Design, 122 (4), 359-370.
15. Vermaas, P. E., Eckert, C., (2013), "My Functional Description is Better!", Artificial Intelligence for Engineering Design, Analysis and Manufacturing, 27, 187-190.