Representation, Realization, and Ascription of Functions for Material Entities

Patryk Burek, Heinrich Herre

Institut für Medizinische Informatik,
Statistik und Epidemiologie (IMISE)
Medizinische Fakultät,
Universität Leipzig
Härtelstrasse 16-18
04107 Leipzig
burek@informatik.uni-leipzig.de
heinrich.herre@imise.uni-leipzig.de

Abstract: Across many domains entities are described in terms of their functionality, i.e. in terms of functions, which are ascribed to them. We propose an ontological framework, called OF (Ontology of Functions), which provides an expressive formalism to specify the central components of a function and to classify various versions of function ascription. OF is an integral part of GFO; it uses heavily various further categories and axioms of GFO. The notion of function, its representation, realization, and ascription is considered as domain independent, being applicable across diverse particular domains. The results are directed towards a unified theory of functions that is viable for the domains of natural science, notably for biology, and for technology.

1 Introduction

In the current paper, we present the approach to the notion of function which is established by the Ontology of Functions (OF) [Bu06]. This ontology is a module of the General Formal Ontology (GFO) ([He06], [He10]). OF is related to various further categories of GFO and uses several of GFO's basic axioms. Functions exhibit important features of entities, and modeling of functions is a relevant technique that is used across many domains, including business modeling, software engineering, artefact design, and biology. We adopt a common-sense understanding of the notion of a function, exemplified by saying that the function of a warehouse is to store goods, or the function of the car is to transport people or goods. The functional model of an entity focuses on the purposes or goals, associated with it, instead of other aspects such as its physical structure, behavior or its history.

Explicitly stated functions help to understand the modularization of an artifact, since it is often driven by functional decomposition. In software engineering, functional description and functional decomposition are essential aspects, in particular in structured methods. Modeling of functions often pertains to the question under what conditions a function can be assigned to an entity. This perspective is motivated by the need of representing the functionality of entities, in particular of artifacts.

In OF, the ascription of a function to an entity is formally expressed by means of a ternary has-function relation HasFu(x,y,z) having the meaning that an entity x, called a function bearer, has a function y, denoted by Fu(y), in context z. This relation, the third argument of which points to a context, corresponds to the notion of function in the interpretation of Cummins [Cu75], who observed that function ascription is relativized to a context. In his understanding, the context of the has-function relation has an epistemological character; it is an analytical account of a capability of some entity in which an object under question is considered. The has-function relation comes in a number of variants, namely as an actual, dispositional or an intended has-function relation. Often, to have a function means to realize (actually or potentially) that function. Therefore, one could identify the has-function relation with the realization relation.

However, in this paper these both relations are distinguished; it is possible that an object has a function although it is not realizing that function. The realization relation is an objective, subject-independent, relation, observable in the world, because it is usually related to a goal achievement, whereas the relation of the has-function, considered independently from any realization, often involves an aspect of subjectivity. The intended has-function relation permits to handle yet another type of function ascription, namely such which is not based on an actual or dispositional realization, but, instead, originates from the intention of an agent.

Typically, functions are assigned to objects that execute realizations of functions, (e.g. in [AS08]). A heart, for example, is playing the role of pumping blood, and it is often said to have a function of pumping blood. However, function bearers are not restricted to objects executing a realization of functions, but can be also associated to processes or other entities. In [CJ00] the authors Chandrasekaran and Josephson emphasize the difference between functions of objects and functions of processes. It is not only a heart that may have a function of blood pumping, but also of the process of a heart's behavior may be said to have that function. In OF, the realization of a function is not necessarily a process, but also could be a structure, a state or a situation. Consider, for instance, the structure of a battledress which realizes a function of a camouflage. Hence, we propose to draw the distinction not between functions of processes and functions of objects – as introduced and elaborated by Chandrasekaran and Josephson- but rather between function realizations and their actors. Thus, in OF a function may be ascribed to a realization and/or to the actor of a realization.

In summary, the ontology of functions OF is characterized by following features:

- It is independent from any special domain, hence, it belongs to a top level ontology (in our case it is part of GFO).
- It provides a level of representation for functions, which is independent from any realization.
- It provides a theory for the realizations of a function.
- It provides a theory for the ascription of functions, based on a ternary relation HasFun(x,y,z).

We believe that any top level ontology of functions should satisfy these conditions, abbreviated by RRA (Representation-Realization-Ascription). The paper is devoted to an explication of the RRA model.

2 Basics of GFO

The ontology of functions OF, being a part of GFO, depends on various entities and axioms which are subsequently summarized. In this paper we restrict to the case that the bearers of functions are material entities, though, our framework can be extended to other types of entities. In GFO the entities of the world are classified into categories and individuals. Categories can be instantiated, individuals are not instantiable. GFO allows for categories of higher order, i.e., there are categories whose instances are themselves categories for example the category “species”. Spatio-temporal individuals are classified into continuants, presentials and processes. Continuants persist through time (cars, balls, trees, ...), processes (temporally extended material entities that happen in time, for example a run), whereas presentials being wholly present at a time-point (this car or this ball at a time-point t). Continuants exhibit at every time-point of its life time a presential. The corresponding classes/sets of individuals, denoted by the predicates Cont(x), Pres(x), and Proc(x), are assumed to be pair-wise disjoint. Processes present the most important kind of entity, whereas presentials and continuants are derived from them. There are several basic relations which canonically connect processes, presentials, and continuants. The integration axiom of GFO states that for every continuant C there exists a process P the boundaries of which coincide with the presentials, exhibited by C ([He06], [He10]). In comparison to other top level ontologies (as BFO [BFO], DOLCE [MBG03], UFO [GW10]), GFO is the only ontology, used in practical applications, for which the processes are the most fundamental category of spatio-temporal individuals, whereas objects and their snapshots (presentials) depend on processes.

GFO provides several types of complex individuals as, for example, situations or situoids. These are parts of reality which can be comprehendet as a coherent whole [BP 1983]. An example of a situoid is a football match with temporal extension and spatial location, which includes various entities such that a coherent whole is established. Situations are restrictions of situoids to time points, for example, a snapshot of a football match. Situations and situoids are considered as individuals, the specification of which needs universals (in particular relational universals), associated to them. We assume that contexts can be regarded as situoids.

Attributives are individuals that existentially depend on other entities as bearers. There is a variety of types of attributives, among them, qualities, roles, and structural features. The bearers of these attributives can be continuants, presentials and processes. Categories whose instances are attributives are called properties. According to the different types of attributives (relational roles, qualities, structural features, individual functions, dispositions, etc.) we distinguish quality properties (or intrinsic properties) and role properties (extrinsic properties), and the role properties are classified into relational role properties (abr. relational properties), social role properties (social properties) etc. Attributives are considered, in a sense, as elementary or atomic entities; the same holds for their categorial analogs/counterparts, the properties. Attributives can be combined to more complex entities, called complex attributives. An individual material object, for example, exhibits a bundle of attributives. The corresponding categories of complex attributives are called complex properties.

3 Representation of Functions

In OF a function possesses a psychological entity as a component. This entity is an idea or a thought which is directed to a goal in the future. Psychological entities belong to the psychological stratum [Po01], being one of the four fundamental ontological regions adopted by GFO. The elaboration of the psychological stratum is work in progress; it uses result from Gestalt-Theorie and cognitive psychology, [Al01), [Al03].

This goal of a function can be understood as a set of situations to be achieved, and this set is described by a category. These goals are imagined in the future (they are, in a sense, anticipated entities); hence, to achieve them, we must start with the existing present situations. These situations, we are starting from, are called requirements. On a more abstract level, a function in OF relates initial situations, called requirements, to situations to be achieved, called goals.

A similar approach is taken by Hartmann [Ha65] which can be summarized as follows. A function exhibits three elements: the setting of a goal in the future, the planning of how to achieve the goal, resulting in an entity that is capable of achieving the goal through causal means. Though, there is a difference between OF-functions and Hartmann's approach. Functions in OF are specified on a level, being independent from any realization.

The plan in Hartmann's approach, the realization of which achieves the goal, belongs in the OF-theory to the level of realization. Functions depend on a bearer and in Hartmann's theory these bearers are entities that are capable of achieving the goal through causal means. The connection of a function to a bearer is determined in the OF-theory by a functional item, being a relevant component of a function's specification.
A functional item is a set of necessary conditions that an entity must satisfy to be able to achieve the goal within a realization. Hartmann's approach describes the general principle, called backward planning, how such a functional item can be found.

The term function in OF exhibits various meanings which are made explicit as follows. A function f can be an intentional entity, called intentional function, specified by the predicate IntF(f); a function f can be understood as a conceptual structure, called conceptual function, and specified by ConcF(f); a function f can be an individual function, specified by IndF(f); finally, a function f can be understood as a universal function, specified by the predicate UnivF(f). There are relations between these different interpretations of a function which can be expressed by logical formulae. We stipulate, for example, the condition that an intentional function is represented by a conceptual function, formally, $\forall x\ (IntF(x) \rightarrow \exists y\ (ConcF(y) \wedge repr(y,x))$; and that the instances of universal functions are individual functions, formally: $\forall xy\ (UnivF(x) \wedge y::x \rightarrow IndF(y))$.

Throughout the remainder of the paper we use the term function in the sense of a conceptual structure, the term universal function as a concept, and the term individual function denotes an individual. In the framework of GFO an individual functions is an attributive, a universal function is a concept, whereas a conceptual structure is a system composed of concepts and sets.

Definition. A function f is a conceptual structure CStr(f) of the following form:
CStr(f)=(Label(f), Req(f), Goal(f), FItem(f)), where:

- Label(f) denotes a set of labels of function f, which are natural language expressions, informally describing "to do something".
- Req(f) denotes a concept, called requirements, the instances of which are parts of material reality, which must be present if the function f is to be realized.

- Goal(f) denotes a concept the instances of which of are parts of material reality being intended by some agent as a result of successful realizations of the function f.
- FItem(f), called functional item of f, is a system of necessary properties which a bearer of f must satisfy to execute a realization of f.

The interpretation of a function f as a conceptual structure CStr(f) raises the question what a conceptual structure is and how it is classified within GFO. A conceptual structure is a system composed of concepts and sets, and is classified under the category system. To any function f, given by CStr(f), we associate a uniquely determined concept UnivF(CStr(f)) the instances of which are individuals, called individual functions. These functions are called actual functions, according to definition 1 (section 5.1). Individual functions are cognitive creations, used to interpret the world, similarly as ontological relations, the instances of which are relators [Lo2009]. We say that a material entity e is preemptive for function f, denoted by preemptive(e,f), if e satisfies the conditions of the functional item FItem(f). The relation preemptive(x,y) connects the function y to a material entity x. If the entity e has an actual function related to f, i.e. if there is an instance of UnivF(CStr(f)) inhering in e, then e is preemptive for f. The converse is not true; an entity e, being preemptive for f, does not necessarily possess an actual function..

The development of a conceptual function-structure (representing an intentional function) involves various non-trivial steps. In the first step a function is often specified informally by a natural language expression of the form to do something, hence, a function is informally represented using a verb and a noun, [Mi72]. For instance, the function of an anti-burglar system could be specified by the following statement: Prevent, detect, notice and react on unauthorized incident. In the second step the goals and requirements are specified. Requirements and goals are understood as parts of material reality, and in the framework of GFO they can be conceived as situations or situoids or parts of them. For these sets of requirements and goals we must create concepts (a goal concept, and a requirement concept), the instances of which include the former mentioned sets.

The construction of a functional item is a crucial and possibly the most difficult one. We must find conditions to be satisfied by entities that enable them, according to our belief, to act as a realizer of the realizations of the function. In artefact design the creation of a functional item is probably included in the act of invention. We believe that there is a relation between function-item-construction and the topics, studied in the TIPS method (TIPS = Theory of Inventive Problem Solving), [AS84], [Al84].

In case of non-artefact entities functional items can serve as explanations for understanding the behaviour of these entities. We see here a relation to theory formation, resulting in theories explaining real world phenomena, notably in the field of biology.

In modern physics no functional aspects are present, because it is based on causal laws. But even physics might contain some (hidden) non-causal (and perhaps functional) aspects; these are related to idealizations, for example the law of inertia.

4 Realization of Functions

A function specifies of what is to be done, whereas the realization of functions exhibit another aspect - it refers to a specification of how a goal of the function comes to reality. A realization of a function is an individual spatio-temporal entity which is based on a binary relation, linking a function with an entity to be brought about. Intuitively, we call an actual realization of the function f an entity which results in an achievement of an individual goal of f in the circumstances satisfying an individual requirement of f, hence, an actual realization connects an individual requirement of a function f with an individual goal of it.

The binary relation $Rl_{act}(x,y)$ has the meaning, x is an actual realization for the function y. A category x is called an actual universal realization of a function y, denoted by $UnRl_{act}(x,y)$, if every instance of x is a realization of y, and furthermore, the instances of x cover the requirements of the function.

The relation CausalLink(x,y,z) connects a realization x, and an entity y, being a part of reality, that satisfies the requirements of the considered function, with an entity fulfilling the goal of the function. Hence, x is an entity that causally connects the entities y and z, which can be briefly described by the condition that z is caused by y by means of the entity x. The relation CausalLink will be taken as a primitive relation. x::y has the meaning that x is an instance of y, and y being an category.

The relation, denoted by x @ y, has the intuitive meaning that an individual x, being a complex whole, fulfills the category y or the individual y in the sense, that an instance of y is a part of x or y is a part of x. For example, the function goal goods are located in Berlin is fulfilled by every situation (being a whole), which contains as its parts the situation of individual goods being located in Berlin. Formally, the relation $Rl_{act}(x,y)$, x is a realization of the function y, is defined as follows:

(1) $Rl_{act}(x,y) \leftrightarrow \exists\, u\, v\, (u :: Req(y) \wedge v::Goal(y) \wedge CausalLink(x,u,v))$

The relation $UnRl_{act}(x,y)$, x referring to a universal realization of the function y, has the intuitive meaning that for every individual requirement s of y there exists an individual goal t of y, and an individual realization u of x such that u causally connects s with t. Since usually only parts of s and t are causally related by u the relation $UnRl_{act}(x,y)$ is formally specified by the condition (2):

(2) $UnRl_{act}(x,y) \leftrightarrow Fu(y) \wedge \forall\, s\, (s :: Req(y) \rightarrow \exists vwtu(t::Goal(y) \wedge v@s \wedge w@t \wedge u::x$ $CausalLink(u,v,w)))$

To illustrate formula (1) take, for example, the function f: to transport goods G from Peking to Berlin in time period T and a process p: transport of goods G by plane from Peking to Berlin in time period T. Then, we can say that p is a causal process which starts with requirements satisfying the corresponding conditions of f and which ends by achieving the goal of f. In this sense, this process p can be called a realization of the function f. The relation CausalLink(x,y,z) is a basic relation in the ontology of causality: z is caused by y by means of the mediating entity x; x can be a process whose initial boundary is y and end boundary is z. But it is also possible that y and z are presentials, existing at different, though coinciding time-points; in this case the entity x consists of this pair (y,z), which is a direct causal link. Furthermore, y and z can be also processes. Direct causal links are treated in [Mi2005], and this theory is included into GFO.

Often a realization is a complex entity which has a number of entities participating in it. In certain situations some of these entities can be identified as those which execute the function realization. The execution of entity y by entity x, denoted by the binary relation Exe(x,y), is understood as a causal influence that x has on y[1]. For instance, to the process p of blood movement, which is a realization of function f: to pump blood, contribute the heart h, the blood, and the veins. However, the role that the heart h has in the process p is different from the roles of veins and blood, namely, it is the heart which actually pumps blood and, thus, it can be said that the heart h executes the realization p of the function f. The inter-relations between the mentioned entities are expressed by the fact Exe(h,p), saying that the heart h executes the process of blood movement p.

An individual x, executing a realization of a function f, is called an actual realizer of function f and is expressed by a relation $R_{Act}(x,f)$. Formally,

(3) $R_{Act}(x,y) \leftrightarrow Fu(y) \wedge \exists q\ (Rl_{act}(q,y) \wedge Exe(x,q))$

We can describe entities not only due to their actual state of being a realization of a function or executing such a realization, but also due to their capability of doing that. For this purpose we introduce the notions of dispositional realizer, denoted by $R_{Disp}(r,f)$, where r is a dispositional realizer of function f, and of dispositional realization, denoted by $Rl_{disp}(x,f)$, where x is a dispositional realization of function f.

For example, if we consider an arbitrary flight of a plane, it is intuitive to judge whether it could be used as an actual realization of the function of transportation or not. In this sense, the process of flight is considered as a potential realization of this function, but not as an actual one, since it does not actually realize this function, but only could do so. Hence, if we consider an individual process p of flight, by which no goods are transported, then the formula 2 does not hold for p. Since p is not an actual realization of the function, but only a dispositional one. Thus, it is not the case that the occurrence of p leads to an achievement of the function's goal, but, rather that there is another process q which is similar to p and which is an actual realization of the function f. To make this precise, we augment the realization level of the function f by a similarity relation sim(x,y,z) which is defined for spatio-temporal entities x and y. The relation sim(x,y,z) has he meaning the entities x and y are similar with respect to the function z. For example, the function to hit nails may be realized by a process of hitting nails executed by an actual realizer, being a hammer h. If an entity g is similar to h then g could as well serve as an actual realizer, though, at present, g might not be used as an actual realizer. The function z must determine this similarity relation sim(x,y,z), in particular, in our example, z will exclude that an elephant e is similar to the hammer h, since an elephant can never be used to hit nails the way a hammer does. This similarity relation is also applied to realizations, and it is a difficult problem to specify the similarity relation between realizations which is appropriate for the function z.
A dispositional realization can, then, be defined as follows

(4) $Rl_{disp}(x,y) \leftrightarrow Fu(y) \wedge \neg Rl_{act}(x,y) \wedge \exists u(sim(u,x,y) \wedge Rl_{act}(u,y))$

We consider an entity x to be a dispositional realizer of a function f if there is an entity z being similar to x with respect to f such that z is an actual realizer of f. Generally we assume that the relation sim(x,y,f) is closed with respect to the functional item FItem(f),

[1]Both the notions of achievement and execution are strongly related to the notion of causality and itself are important and are non trivial problems which however are out of the scope of the current paper and thus taken here as primitive.

hence, if x satisfies FItem(f), and sim(x,y,f), then y satisfies FItem(f), too. Furthermore, we assume that any actual realizer of the function f satisfies the functional item FItem(f).

Then, x is a dispositional realizer of the function y if x not an actual realizer of y, though there exists an entity z, similar to x with respect to y, and z is an actual realizer of y. From this follows that x satisfies all conditions specified by the functional item of y, but it is not an actual realizer of y. Formally,

(5) $R_{Disp}(x,y) \leftrightarrow Fu(y) \land \neg R_{Act}(x,y) \land \exists z\,(sim(z,x,y) \land R_{Act}(z,y))$

For example, if we consider an arbitrary plane, it is intuitive to judge whether it could be used as a realizer of the function of transportation by evaluating such its properties as e.g. capacity.

An entity, to which a function is assigned by means of the has-function relation we call a function bearer. The ontology OF admits that functions can be ascribed, among others, to processes, to objects, to situations, and to quality values.

5 Ascription of Functions

In this section we present an ontological analysis of the notion of function ascription which is expressed by the relation HasFu(x,y,z) with the meaning the entity x has the function y in context z. The relation HasFu(x,y,z) is considered as a basic relation which has an intuitive, non-formal, meaning. The following sentences exemplify the HasFu relation: an engine in context of a car has a function of generating rotating motion; a hammer in context of a pile of papers on a desk has a function of preventing them from being blown by the wind; a heart in the context of a human body has the function to pump blood.

The context is understood in a broad sense, covering also contexts which exhibit spatio-temporal parts and surroundings. In this reading, a desk x standing in a room y can be considered to have the room y as a context; a heart, being a part of a body, has the body as a context; an animal may have a niche as a context; a football game may have the stadion as a context. In all these cases the considered entity is a component of a part of reality, which, in turn, is a temporal-spatial part. The relation between an entity x and a context is y expressed by HasContext(x,y).

It turns out, that there are several versions of HasFu-relation and for any of these relations $HasFu_j(x,y,z)$ we introduce the axiom : $HasFu_j(x,y,z) \rightarrow HasFu(x,y,z)$. The family of HasFu-relations is classified into actual/dispositional has-function relations and in intended functions-relations. The intended has-function relation is classified into several further subcategories.

5.1 Actual and Dispositional Functions

We say that an individual a is preemptive for the function f if a satisfies the functional item of f. This satisfaction relation connects the function with a bearer.

Definition 1 (Actual Function). An individual x has the function y as an actual function in context c, denoted by $HasFu_{Act}(x,f,c)$, iff x is an actual realization of y in c, or x is an actual realizer of y in c.

(1) $HasFu_{Act}(x,y,z) \leftrightarrow Fu(y) \wedge HasContext(x,z) \wedge (Rl_{act}(x,y) \vee R_{Act}(x,y))$

This is the most straightforward type of function ascription. For example, an object realizing some goal in a given context may be considered by some agent to have a function to realize this goal. The above definition reflects the intuitions of the dual character of function ascription - a function may be ascribed either to a realization or to a realizer. For example, not only a flying plane can be considered to have a function of transporting goods, but also a process of flight.

Frequently, an item has a function ascribed although it does not actually realize this function. For example, a plane waiting at the airport has a function of transporting goods although it is not realizing it in this given situation.

We recognize several kinds of non-actual function ascriptions. Firstly, we will introduce the dispositional has-function relation. Just as in the case of the actual has-function, the dispositional has-function is defined both for realizations and actors of realizations (realizers).

Definition 2 (Dispositional Function). An individual x has the function y as a dispositional function in context c, denoted by $HasFu_{Disp}(x,y,c)$, iff x is a dispositional realization of y in z or x is an actor of a dispositional realization of y in z.

(2) $HasFu_{Disp}(x,y,z) \leftrightarrow Fu(z) \wedge HasContext(x,z) \wedge (Rl_{disp}(x,y) \vee R_{Disp}(x,y))$

If a flight from A to B is a dispositional realization of transporting goods, then it follows that it has a dispositional function of transporting goods. Analogously, a plane being a dispositional realizer has a dispositional function to transport goods.

5.2 Intended Functions

The common and intuitive interpretation of the statement "X has a function F" is "X was designed to have a function F". These intuitions are reflected by several approaches e.g. [Bo92]. However, from the statement that an item was designed to realize a given function, does not follow that the item actually realizes this function or that it has a disposition to realize this function. As examples we consider broken or ill-designed artifacts.

Therefore, function ascription, founded on the designer's intentions, cannot be reduced to the actual or the dispositional has-function relation and should be considered as a third kind of function ascription.

On the other hand, if one agrees on function ascription based on a designer's intention, then the question could be raised why to take only the designer's intentions into account. Apart from a designer, the process of artifact construction also involves other parties that have an influence on the function of an artifact. Let us take the example of software engineering process. Most typically, the process starts with the requirements of stakeholders. On the basis of their requirements the software is designed by analysts and architects and developed by developers. If software does not meet the stakeholders' requirements, then we could say that it does not perform its function. It may be the case that software is ill-designed: intentions of designers do not reflect the requirements, or the software is ill-developed: designers' intention may meet the requirements but they may be inappropriately implemented.

Therefore, it seems that not only the intentions of designers have an influence on the artifact's function, but other parties like stakeholders should also be included. Thus, the statement that the function of an item is what it was designed for seems to be too narrow, since the function of an item may be not what an item was designed for, but what an item was expected to do.

We introduce a third kind of function ascription, namely the intended has-function relation. This relation subsumes the intentions of designers, stakeholders and other parties in the assignment of functions.

Definition 3 (Intended Function). An individual x has an intended function y in context z, (x has y as an intended function in z), denoted by $HasFu_{Inten}(x,y,z)$, due to some agent q who intends x to have the function y in context z. Formally,

$$(3)\quad HasFu_{Inten}(x,y,z) \leftrightarrow \exists qvr\ (Agent(q) \wedge Intent(q,v) \wedge IntCont(v,r,x,y,z) \wedge r::HasFu)$$

The relation Intent(a,i) has the meaning that an agent a intends i. The content of the intention i is depicted by the predicate IntCont(i, R, $a_1...a_n$) where R is an n-place relation and $a_1, \dots, a_n$ are arguments of R. In the above definition the entity r is an instance of the has-function relation HasFu, interpreted as a category, hence r is a relator, [He10]. Thus, herein the content of the belief is "x has function y in context z".

Formula 3 has the drawback because it makes function ascription too general and too liberal. Not all agents, who intend an item to have a function, have the power to ascribe functions to items. The statement that an artifact has function f is not treated equally when expressed by an artifact's designer or by a person having no particular knowledge about the artifact.

This shows that not every agent's intention should be considered when ascribing functions to items but rather some agents are more reliable in ascribing functions than others. Some agents, or, to be more precise, some types of agent roles, are of particular importance for the ascription of functions to artifacts. These are: stakeholder, designer, user and researcher roles. The role players of these roles are not disjoint, thus one agent can play several of them.

Each of these roles reflects a different interest in an artifact. A stakeholder is an agent, whose requirements should be satisfied by the artifact, and which motivate the manufacturing of the artifact. A function required by a stakeholder x of an artifact y, denoted by Stakeholder(x,y), is called a required function of an artifact and is denoted by the predicate $HasFu_{Req}(x,y,z)$ having the reading that x has a required function y in context z.

Definition 4 (Required Function).

$$(4)\quad HasFu_{Req}(x,y,z) \leftrightarrow \exists qvr\ (Stakeholder(q,x) \wedge Intent(q,v) \wedge IntCont(v,r,x,y,z) \wedge r::HasFu).$$

A designer is an agent x that designed an artifact y, denoted by the expression Designer(x,y). A function designed by a designer is called a designed function and is denoted by $HasFu_{Desig}(x,y,z)$, having the reading: x has a designed function y in context z.

Definition 5 (Designed Function).

$$(5)\quad HasFu_{Desig}(x,y,z) \leftrightarrow \exists\ qvr(Designer(q,x) \wedge Intent(q,v) \wedge IntCont(v,r,x,y,z) \wedge r::HasFu).$$

A user x of an entity y, denoted by the relation User(x,y) , is an agent x, who uses an artifact or some other entity y. A function intended by a user is called a user function and is denoted by the expression $HasFu_{User}(x,y,z)$, having the reading: x has a user function y in context z.

Definition 6 (User Function).

(6) $HasFu_{User}(x,y,z) \leftrightarrow \exists\, qvr(User(q,x) \wedge Intent(q,v) \wedge IntCont(v,r,x,y,z) \wedge r :: HasFu)$.

Finally, a researcher x is agent x who examines an artifact or some other entity y. The relation between a researcher x and an artifact y is denoted by the expression Researcher(x,y). A function assigned to an item by a researcher is called a researched function. The researched function is formally captured and denoted by the relation $HasFu_{Res}(x,y,z)$, having the reading: x has a researched function y in a context z.

Definition 7 (Researched Function).

(7) $HasFu_{Res}(x,y,z) \leftrightarrow \exists\, qvr(Researcher(q,x) \wedge Intent(q,v) \wedge IntCont(v,r,x,y,z) \wedge r :: HasFu)$.

The intended functions are only the matter of intentions of particular agents and do not involve actual nor dispositional realization. Therefore, none of the above function ascriptions guarantees that the software really realizes any of those functions.

An item has an intended function in a situation s iff it is intended to have a function in s. The question, then, is what does it mean that an agent intends an item to have a function in a given situation? We distinguish three possibilities, according to the three kinds of the has-function which have been distinguished. In the first two cases an item has a function in a given context, because an agent intends that an item actually realizes, or has a disposition to realize the function in that context. In these cases an agent intends an item to have an actual or resp. dispositional function. In the third case an agent intends that an item has an intended, if only by some other agent, function. Take as an example an archaeologist exploring an ancient artifact. The archaeologist may have reasons to claim that the artifact was designed to perform a given function. In this case a function that is assigned to an artifact is an intended-intended function. The researcher intends that a designer intended that the artifact should have a function f in s. Note, that the artifact may be wrongly constructed so that not only the artifact never actually realized that function, but also it never had a disposition to realize it. This, however, does not disturb the archeologist's claim concerning the intended function. The intended-intended function is independent of an actual or a dispositional function.

6 Applications

The interpretation of what it means to apply a formal ontology (as a theory) needs further explanation. A theory can be applied to another theory, and if these theories are located at the same level of abstraction, we call them horizontal applications. But often application of a theory is understood in the vertical direction, from the abstract to the concrete. The idea of a network of application links between areas of different levels of abstraction was formulated in the programmatic paper of Mises [Mi21]. Mises claims that in such a network every relevant level of abstraction must be present, otherwise the progress of science and its applications will be hampered. The ontology of functions (OF) is intended to be applied in horizontal as well as in vertical direction.

One aim of OF is the development of a foundation for central categories which should be included in every top level ontology. An ontology of functions must be used to establish and formally axiomatize the domain specific knowledge of areas in which functions play an important role. We hold that foundational research should uncover and introduce new ideas, whereas technical applications should be grounded on a clear and well-established conceptual basis.[2]

The hitherto applications of the OF pertain to the field of bio-ontologies [BLH06] and software and system modeling [BHL09]. In [BHK06] it has been demonstrated how to use the OF to represent the relation between biological processes and functions in the Gene Ontology [As00], for which no ontologically founded representation formalism is currently available. In [BHL09] is presented the applications of OF for the clarification of basic notions used in system and software modeling such as functional decomposition.

7 Related Work and Discussion

There is a rich computer science literature on functions and functional modeling, especially in context of functional device representation. The main assumption of functional device representations which contrast with the OF is that a function is considered as a domain notion and not as a general one, and often a function is ascribed only to physical objects or even technical devices (e.g. Functional Concept Ontology (FCO) [KSN02], [KM04], Functional Representation framework (FR) [Ch94], Functional Representational Language (CFRL) [IVF95]. The exception is the ontological framework of Chandrasekaran and Josephson [CJ97]. However, the direct application of the notion of function - acceptable in device representation - to other domains results in oversimplification or in unnecessary teleological assumptions.

Many works show a strong correlation between the notions of function and behavior. For example in FR, CFRL, FCO, Function Behavior State, see [UTT90], [UT95], the notion of function is grounded in the notion of behavior which makes it impossible to speak about functions independently of the behavior realizing it.

The definition of function in behavioral terms reduces its scope to so-called dynamic functions involving actions and changes of the object on which they operate and raises problems with the representation of passive functions [Ke89], which are not related to dynamic behavior. Moreover, it does not permit to speak about functions independently of their behavioral realizations, which, however, is required in many situations, e.g. in the early phases of the design. Finally, functions defined in terms of behavior impose a particular behavioral realization, which make them realization-dependent and prevent from handling alternative realizations.

In OF we do not restrict function bearer to any particular ontological kind thus e.g. we follow the intuitions of Chandrasekaran, Josephson [CJ97] that not only objects but also processes may have functions. For example, the process of boiling water may have the function to produce steam.

[2] Hamming [Ha97] "In science, if you know what you are doing, you should not be doing it. In engineering, if you do not know what you are doing, you should not be doing it."

The problem of conditions for function ascription in device representation literature is treated diversely. For instance, in Function-Behavior-Structure framework [Ge90], [QG96] a function is considered as everything an object does, in [Li94] - as intended by a designer role of an entity, in [CJ97] as what an object does and is intended by a designer or user to do, analogously.

OF is designed as a domain independent top level ontology. Other top-level ontologies do not include the notion of function, or treat it cursorily, only. Descriptive Ontology for Linguistic and Cognitive Engineering (DOLCE) [MBG03] and Sowa's ontology [So00] lack a developed and established notion of function or any correlated notion. The Suggested Upper Merged Ontology (SUMO) [PN02] introduces the hasPurpose relation, which has the meaning that a physical thing has a desired or expected purpose. The notion of a purpose is distinguished from the notion of an outcome, which does not need to be expected or desired. The intended purpose in SUMO could be interpreted as a function of an entity. However, purposes in SUMO are assigned to physical objects only. In Sowa's ontology the concept of purpose is also present, however it is not considered as a function but as the relation gluing an agent, his act and his intention concerning that act.

The Basic Formal Ontology (BFO) [BFO06] provides a notion of function defined as "a realizable entity the manifestation of which is an essentialy end-directed activity of a continuant entity in virtue of that the continuant entity being a specific kind of entity in the kind or kinds of contexts that it is made for" ([BFO06 p.55]), where realizable entities are "dependent continuants that inhere in continuant entities and are not exhibited in full at every time in which they inhere in an entity or group of entities" ([BFO06] p. 53). From the above formulations one can conclude that in BFO
(1) a function is defined by the references to its manifestation, i.e. realization which is a teleological process and (2) function is a contextual notion. Hence, a function in BFO is a realization dependent notion. Additionally, only functions realized by processes (activities) are handled. In contrast, a function in OF as such is defined independently from its realization by a conceptual structure and function realizations are not restricted to processes only. Neither is the notion of function in OF considered as contextual. Furthermore, BFO makes a rigorous restriction on the types of entities, which can have functions ascribed. These are organisms, certain tool-like products of organisms, intentionally designed artifacts, and certain kinds of social groups or institutions. In contrast, OF is more expressive making no restriction on function bearers; these can be enduring entities but also processes or even situations.

In [MKB12] ideas on a unified definition of function are discussed. On the one hand, that paper comprises various notions surrounding functions, though with somewhat different notation, that have already been discussed in detail in [Bu06] (which the authors do not seem to be aware of), which leads to some similarity to the approach presented here. On the other hand, there are also some differences. For instance, in [MKB12] there is no strict separation between a function as conceptual structure and the realization of a function. Furthermore, in contrast to OF a realization is always represented by a behaviour, being a process.

8 Conclusions and Future Work

In the current paper we have investigated the notion of function ascription and have introduced a number of notions that permit to model it not only for artifact systems but also for natural systems. Several types of function ascription have been recognized, among them the intended has-function, typical for artifacts, and the actual and the dispositional has-function, founded on the notions of actual and dispositional realizations applicable to non-artifacts as well. The notions introduced permit to ascribe functions to arbitrary entities, though we restricted to material entities, including material continuants, material presentials, and material processes.

The presented above analysis is a part of OF which aims on providing a framework for modeling functions and functional knowledge. The framework is organized around the following issues: how to model a separated function, how functions are interrelated and organized into functional models, how functions can be related to non-functional entities, and finally how functions can be incorporated into the broader ontological framework. We expect extensive practical applications of our framework in diverse domains.

The clear understanding of function ascription permits for applying OF for definition and specification of biological functions, and their relation to other entities in biology. Application of OF prevents errors, and helps in clarification of definitions.

According to our understanding the top level ontological analysis, as the one reported in the current paper, can provide a foundation for conceptual modeling and support knowledge engineers in construction of domain knowledge bases. As presented in [Gu05], they can be used for the development of UML profiles.

The analysis presented in the current paper pertain to the understanding of the notion of function relations which are the fundamental relations linking non-functional entities with functions. Thus the presented analysis are in our opinion of importance for linking of functional models with non-functional, e.g. behavioral or structural, models.

The presented results require further work in particular the next task is the development of a full axiomatization that specifies the considered relations and categories. The current paper presents a fragmented axiomatization only. The first step of this task is the explication of the primitive relations which are assumed. For these relations axioms must be invented and introduced. The remaining relations and predicates are introduced by explicit definitions, many of them are already introduced and discussed in the present paper.

Acknowledgments

We are grateful to Robert Hoehndorf, Frank Loebe, and Roberto Poli for inspiring discussions on the topics presented in this paper.

References

[AS84] G. Altschuller, A. Seljuzki: Flügel für Ikarus - Über die moderne Technik des Erfindens. Verlag MIR Moskau und Urania-Verlag, Moskau 1983, 292 Seiten

[Al84] G.S. Altschuller: Erfinden – Wege zur Lösung technischer Probleme. VEB Verlag Technik, Berlin 1984. Limitierter Nachdruck 1998, 280 Seiten,

[AS08] Arp, R., Smith, B. (2008). Function, Role, and Disposition in Basic Formal Ontology Proceedings of Bio-Ontologies Workshop (ISMB 2008), Toronto, 45-48.

[AB00] Ashburner, M., Ball, C. A., Blake, J. A., Botstein, D., Butler, H., Cherry, J.M., Davis, A. P., Dolinski, K., Dwight, S. S., Eppig, J. T., Harris, M. A., Hill, D. P. , Issel-Tarver, L., Kasarskis, A., Lewis, J S.. Matese, C., Richardson, J. E., Ringwald, M., Rubin, G. M., Sherlock, G.: Gene ontology: tool for the unification of biology. The Gene Ontology Consortium. Nat Genet, 25(1):25–29, May, 2000.

[Al01] Albertazzi, L. 2001. Presentational primitives. Parts, wholes and psychophysics. In Early European Contributors to Cognitive Science, ed. L. Albertazzi, 29–60. Netherlands: Kluwer.

[Al03] Albertazzi, L. 2003. From Kanizsa back to benussi: Varieties of intentional, reference, Axiomathes, 13:3–4, 239–259.

[BFO] http://www.ifomis.org/bfo

[Bo92] Bonnet, J. C.: Towards a formal representation of device functionality. TR 92-54, Knowledge Systems Laboratory, Stanford University, 1992.

[BHL06] Burek, P., Hoehndorf,R., Loebe,F. Visagie, J., Kelso, J., Herre, H. A top-level ontology of functions and its application in Open Biomedical Ontologies. Bioinformatics Vol. 22(14), pp. e66-e73.

[BLH09] Burek, P., Herre, H., Loebe, F. Ontological Analysis of Functional Decompostions. Proceedings of the 8th Interentional conference of Software Methodologies, tools and techniques. SOMET-2009

[BHR10] Burek P, Herre H, Roeder I, Glauche I, Scherf N, Lo¤er M. Towards a Cellular Genealogy Ontology. In: Proceedings of the 2nd Workshop on 'Ontologien in Biomedizin und Lebenswissenschaften' (OBML), Mannheim, Germany; IMISEReports 2/2010 pp. 59-63. Mannheim; 2010. p. 59-63.

[BP83] Barwise, J., Perry. J. Situations and Attitudes. Cambridge- MA: MIT Press, 1983

[Ch94] Chandrasekaran B.: Functional Representation and Causal Processes. Advances in Computers, 38: 73-143, 1994.

[CJ97] Chandrasekaran, B., Josephson, J.R.: Representing Function as Effect. Proceedings of the Functional Modeling Workshop, Paris, France, 1997.

[CJ00] Chandrasekaran, B., Josephson, J. R. : Function in Device Representation. Engineering with Computers, Special Issue on Computer Aided Engineering, 16:162-177, 2000.

[Cu75] Cummins, R.: Functional Analysis. Journal of Philosophy 72:741–765, 1975.

[Ge90] Gero, J.S.: Design prototypes: a knowledge representation schema for design. AI Magazine, 11(4): 26-36, 1990.

[Gu05] Guizzardi, G.: Ontological Foundations for Conceptual Models. PhD Thesis, Telematics Instituut fundamental Research Series, No. 015, Telematics Instituut, Enschede, The Netherlands, 2005.

[GW10] Guizardi, G., Wagner, G. Using the Unified Foundatioal Ontology Ontology (UFO) as a Foundation for General Conceptual Modeling Languages. In: Poli, R. & Obrst, L. (ed.) Theory and Applications of Ontology. Vol. 2, Berlin: Springer, 2010.

[Ha97] Hamming, R. W. (1997). Art of Doing Science and Engineering. T and F STM.

[Ha66] Hartmann, N. Teleologisches Denken, Walter de Gruyter; 1966.

[He10] Herre, H. General Formal Ontology (GFO) : A Foundational Ontology for Conceptual Modelling. In: Poli, R. & Obrst, L. (ed.) Theory and Applications of Ontology. Vol. 2, Berlin: Springer, 2010.

[IVF95] Iwasaki, Y., Vescovi, M., Fikes, R., Chandrasekaran, B.: A Causal functional representation language with behavior-based semantics. Applied Artificial Intelligence, 9(1):5-31, 1995.

[Ke89] Keuneke, A.: Machine Understanding of Devices: Causal Explanation of Diagnostic

Conclusions. PhD thesis, The Ohio State University, 1989.
[KSN02] Kitamura, Y., Sano, T., Namba, K., and Mizoguchi, R.: A Functional Concept Ontology and Its Application to Automatic Identification of Functional Structures. Advanced Engineering Informatics,16(2):145-163, 2002.
[KM04] Kitamura, Y., Mizoguchi, R.: Ontology-based systematization of functional knowledge. Journal of Engineering Design, 15(4):327-351, August 2004.
[Lo07] Loebe, F. (2007). Abstract vs. social roles – Towards a general theoretical account of roles. Applied Ontology 2(2):127–158.
[Li94] Lind, M.: Modeling Goals and Functions of Complex Industrial Plants. Applied Artificial Intelligence, 8: 259-283, 1994.
[MBG03] C. Masolo, Borgo, S., Gangemi, A., Guarino, N., Oltramari, A.: WonderWeb Deliverable D18. Ontology Library (final). Version 1.0. 31.12.2003. Laboratory For Applied Ontology, ISTC-CNR, Trento, Italy, 2003.
[Mi21] Mises, R. v. (1921). Über die Aufgaben und Ziele der angewandten Mathematik. Zeitschrift für Angewandte Mathematik und Mechanik, vol. 1:1–15.
[Mi72] Miles, L. (1972) Techniques of Value Analysis and Engineering, McGraw-Hill, New York, USA, 1972
[Mi05] Michalek, H. 2005. A Causal Relation Based on Regularity and Manipulability. In: Guizzardi, G. & Wagner, G. (ed.) Proceedings of the EDOC International Workshop on Vocabularies, Ontologies and Rules for the Enterprise (VORTE'05), Enschede, The Netherlands, Sep 20. CTIT Workshops Proceedings, pp. 1-7. Enschede (Netherlands): CTIT.
[MKB12] Mizoguchi, R, Kitamura, Y.,Borgo, S. Towards a Unified Definition of Function. In: Formal Ontology in Information Systems (FOIS), Ed. M Donnely, G, Guizzardi, 2012, p.103-116
[PN02] Pease, A., Niles, I.: IEEE Standard Upper Ontology: A Progress Report. The Knowledge Engineering Review, 17(1):65-70, 2002.
[Po01] Poli, R. The basic Problem of the Theory of Levels of Reality. Axiomathes 2001; 12: 261-283
[So00] Sowa, J.: Knowledge representation - Logical, Philosophical and Conceptual Foundations. Brooks/Coole, Pacivic Grove, USA, 2000.
[UTT90] Umeda, Y., Takeda H., Tomiyama T., Yoshikawa H.: Function, Behavior and Structure. Applications of AI in Engineering, *AIENG'90*, p. 177-193, Computational Mechanic Publications and Springer Verlag, 1990.
[UT95] Umeda, Y., Tomiyama, T.: FBS Modeling: Modeling scheme of function for conceptual design. International Qualitative Reasoning Workshop Proceedings, p. 71-77, University of Amsterdam, the Netherlands, 1995.
[QG96] Qian, L., Gero, J.S.: Function-behaviour-structure paths and their role in analogy-based design. AIEDAM 10:289-312, 1996.