Functional Modelling in Engineering: A First Comparison of Approaches on Two Problems

Compagno Francesco[1,2,*], Borgo Stefano[1]

[1]ISTC-CNR Laboratory for Applied Ontology, via alla cascata 56/C, 38123, Povo, Italy
[2]Adige S.P.A, via per Barco, 11, Levico Terme, 38056, Italy

Abstract
Functional modelling is a well-studied topic in engineering and applies more broadly to areas like biology, management and organization studies. While it is an essential part of building and describing engineering systems, to apply functional modelling consistently in practice remains difficult. This obstacle is especially painful since digitalisation is becoming an increasingly important aspect of modern engineering enterprises. To study the causes of such a difficulty, we analyze two typical problems that are encountered when modelling engineering systems and that are intertwined with functional aspects: identity of system components across time, and granularity management of the system models. The problems are introduced and used to compare a few ontological and engineering approaches that can deal with them. We illustrate our findings with a brief example.

Keywords
Functional modelling, Functional decomposition, Replacement problem, Granularity

1. Introduction

Engineering is about the design and construction of machines, systems and infrastructures, their management and maintenance, including their dismantling at the end of life. The things that engineers create interact with our physical world to obtain desired changes and to prevent undesired happenings. Either way, these things have a reason to exist or, as engineers say, have a function.

Functionality is an essential concept in engineering, and functional modelling has received widespread attention in the literature. Unfortunately, though several theories of functionality have been proposed and studied for more than twenty years, no unique treatment has emerged, and some problems persist. In particular, consistent application of functional modelling is known to be difficult in practice [1]. In this paper, we describe some of the problems that current theories face, and show, using a brief use case, possible ways to tackle them through applied ontology principles.

FOMI 2022: 12th International Workshop on Formal Ontologies meet Industry, September 12-15, 2022, Tarbes, France.
*Corresponding author.
francesco.compagno@loa.istc.cnr.it (C. Francesco); stefano.borgo@cnr.it (B. Stefano)
https://www.istc.cnr.it/people/stefano-borgo (B. Stefano)

2. A Brief Look at the Literature

Functionality is a well-studied topic in engineering and the literature is large, see for instance [2, 3]. We briefly recall the main characteristics of some theories and methodologies that have been developed:

- **Functional Basis (FB)** [4]. Inspired by the German school [2], this methodology considers functions as black-boxes that operate transformations on inputs, producing outputs. FB consists in a vocabulary for functions (e.g., ‘convert’, ‘distribute’) and a classification of inputs/outputs (called flows, e.g., ‘pressure’, ‘electric energy’). The vocabulary is organized in three levels of increasing specialisation. A functional model is a graph whose nodes are boxes labeled with function terms, and whose edges are arrows labeled with flow terms.
- **Functional Representation (FR)** [5, 6]. This methodology builds a structural model of the system at the desired level of granularity. The system is modelled as a set of devices, each one with a set of relevant parameters and input/output ports, and relations between devices, such as the connection between two ports or the containment of one device into another. State variables are attributed to the system and used to determine different system states. Lastly, the system behaviour is represented as a set of state transitions, each one justified either by a ‘causal process description’ (CPD), a ‘domain law’, or a component ‘function’. CPDs and functions can then be justified by other CPDs or functions, forming an in-depth explanation of the system behaviour. Therefore, FR functions are justifications for state transitions. Moreover, FR functions have been defined as intended constraints about the behaviour of a device when isolated (‘device-centred functions’) or when deployed in a given environment (‘environmental-centred functions’). In the latter case, functions are also considered roles in the context of the device environment.
- **Function and Behaviour Representation Language (FBRL)** [7, 8]. This framework starts from a structural model of the system (analogous to the structural models of FR) obtained via a separate module called ‘device ontology’. In FBRL a function is a role that a device behaviour plays in the context of the system, to contribute to the achievement of predetermined goals. The separation between behaviours and functions explains how a device can have different functions despite its behaviour remaining the same (e.g., a heat exchanger can either cool or warm a liquid depending on the system configuration). Moreover, the vocabulary of functional terms (such as ‘to join’) is distinguished from the vocabulary of ‘way-of-achievements’ (such as ‘welding’, a possible way to realize the ‘to join’ function). In this approach, functions are domain-independent, and way-of-achievements can be collected in domain-specific databases. Finally, to give a teleological explanation of a system, a set of ‘meta-functions’ is introduced (e.g., the give-heat function of the boiler *drives* the rotate-shaft function of the turbine).

Several other theories and methodologies exist, the interested reader can see, for example, [9, 10], or the behavioural diagrams of UML or SysML, though the latter do not differentiate clearly between behaviours, functions, and related concepts. There are also theories of functionality developed by the philosophical community, e.g., Cummins [11] and further works in upper ontologies (where FBRL originates) such as BFO, GFO, YAMATO and DOLCE (see e.g. [12, 13, 7, 14], respectively) These theories preicate different ontological categorisations of functions. Here we collect those that we will use later:

- **Functions as dispositions** [15, 16]. According to this view, a function is a disposition that an object manifests provided it has the appropriate physical make-up and the situation satisfies suitable conditions. In this sense, functions inhere in objects, like properties.
- **Functions as transformative events** [17, 18, 19]. Some authors describe classes of functions as particular classes of occurrents. For example, the Functional Basis function 'to convert' is interpreted as the class of all events in which a flow is converted (e.g., whenever a combustion engine consumes some fuel to move a vehicle, that conversion of chemical energy to kinetic energy is a function of 'to convert' type).
- **Functions as role-concepts** [7, 20]. In this view, functions are (or are related to) particular roles contributing to the achievement of some goal. More precisely, in the view of Sasasjima et al. [7] functions are roles of devices behaviours, contributing to some system goal. Instead, for Burek et al. [20] functions are quadruples composed of identifiers, requirements, functional items, and goals. Requirements and goals are state descriptions, respectively preceding and following the realization of a function, while functional items are the roles played by the entities realizing the function.

The modelling of engineering systems encounters many difficulties, in both practical and theoretical aspects. Theories of functionality, as an important part of systems modelling, intersect some of these problems and can help to solve them. In this paper, we focus on two of them, namely, diachronic identity and granularity.

3. Two Problems in Functional Modelling

3.1. Diachronic Identity

Engineering systems and their components are difficult to characterise ontologically. In this section, we argue how an appropriate theory of functions can lessen this difficulty. In particular, we focus on the so-called replacement problem [21], [22, Chapter 14]:

(pr1) **The replacement problem.** In maintenance, the replacement of a malfunctioning component with another is a common occurrence. Thus, it happens frequently that a (sub)system has many or even all of its parts changed over time, but for domain experts that one remains the same (sub)system at all times.

In our opinion, this problem is deeply intertwined with the following, which we call 'the malfunctioning problem':

(pr2) **The malfunctioning problem.** In engineering, the functionality of components/systems is an essential property (indeed, it is the very reason components are employed in a system).

Despite this, components/systems do malfunction and when they do, their identity is not lost.[1]

A solution to this problem is arguably the second desiderata of Varmaas and Houkes in [23], which asks for the 'ascription of proper functions to malfunctioning artefacts'. An obvious idea to solve Problem (pr1) is to introduce in the domain of discourse 'stable' [24], 'conventional' [21], or simply 'system' [22] components as entities that, by definition, survive replacement. This is the way ISO 15926-14 [25] solves the problem: it introduces a category of objects, called 'functional objects' (sometimes 'functional locations'), which is distinct from the category of physical objects. A physical object is a mere constituent of a functional object in the sense that the first can be 'installed-in' the latter. When a component is replaced, the functional object remains the same, only its constituent changes. Furthermore, this approach may solve Problem (pr2) as well, since the function would be ascribed to the functional object (the 'stable' component) as its essential property, while only temporally ascribed to the physical component.

Unfortunately, to implement this strategy, one should characterise these 'stable' entities from the ontological viewpoint, and this turns out to be difficult. ISO 15926-14 proposes a functional characterisation. Other ways to characterise 'stable' components are via physical features and structural relations [24], or introducing a four-dimensional point of view [22]. Additionally, ISO 81346 [26] speaks about 'functional', 'product', and 'location aspects' of a system component in a way that suggests that each of these aspects (or any combination of these) could be used instead of, or in combination with, the functional one.

In this paper we investigate the functional characterisation of system components, for three reasons: first, we aim to show that such a choice is not possible and effective. Second, this choice elegantly solves both problems (pr1) and (pr2), essentially by looking at malfunction as non-conformity to a nominal function (cfr [27], which describes a similar strategy). Third, such an approach relies on already existing engineering methodologies and is, arguably, preferred in some engineering domains (cfr., e.g., the functional objects of ISO 15926).

In the following, we argue that not all ontological theories of functions are equally adapted to be used for such a modelling strategy. To show this, we discuss the distinctions introduced in Section 2:

- **'Stable' entities when functions are dispositions**. This view does not fit well with the previously described strategy, since it does not separate the functions from the physical objects that function. Additionally, engineering functions are, arguably, externally grounded (e.g., depending on the context a heat exchanger can realize a heat or a cool function), while dispositions are not. The view that functions are dispositions is analyzed and criticized also in more general settings, see, e.g., [28].

- **'Stable' entities when functions are transformative events**. This view is not appropriate for our strategy, since we would have at our disposal only individual functions, equated to single 'functionings', that is, individual events during which functions are carried out, and the classes of such events. Neither the classes nor the single individual occurrences could be used as 'stable' entities, because the individual occurrences have

[1]We state the problems in their generality, the reader should bear in mind their practical relevance: a knowledge base that fails to address them, may give wrong answers to queries such as "how many times was the engine replaced?".

limited time extension and one can not predicate on classes in first order logic. One could insist by, e.g., reifying event classes and using these new individuals as stable entities, but there seems to be no natural way to ontologically categorize such reifications.

- **'Stable' entities when functions are role-concepts**. This view seems quite flexible since it relies on an ontological entity, the role, which by construction survives the replacement of its player. This suggests that the approach may satisfy the requirements of the strategy. In the following we investigate it further.

Using functions as role-concepts, to solve problems (pr1) and (pr2) one should identify the entities that survive replacement with functions (i.e., role-concepts), and the entities that don't survive with the physical objects that execute the functions[2]. An objection to this could be raised, stating that 'conventional components', 'functional objects', and the like are entities of a different type than functions. This is the case in, say, [22, 21, 25]. For example, in ISO 15926-14, the entities surviving replacement are 'functional objects', which form a different category than functions and have their own mereology. West [22] explicitly observes that system components are not simply functional objects or roles. The modelling choice of using additional entity kinds, as done in ISO 15926-14, can surely work and solve problems (pr1) and (pr2), but complicates both the ontology and the knowledge information system, and seems to be needed only because of the limitations of the function theory adopted in that system. In fact, one fundamental issue of functional modelling in engineering is functional decomposition, which could be defined as the determination of the teleological (functional) aspects of a system, at increasingly finer levels of granularity. The result depends on the precise methodology used, but is essentially a tree whose nodes are functions and whose edges are interpreted as some kind of part-whole relation.[3] The functional decomposition of a system becomes redundant when one introduces the decomposition of the system into functional objects, for instance through the ISO 15926-'hasFunctionalPart' relation, or any analogous relation[4]. Indeed, if ISO 15926-14-functional objects are not physical objects, what properties distinguish them from functions? Since the relevance of functional decomposition is broadly recognized in engineering and has a rich history in the literature, the use of functions suggests itself as the most direct and natural option. The introduction of additional entities in the domain of discourse may be unnecessary if the goal is just to solve problems (pr1) and (pr2). We do not rule out the possibility that additional entities could be necessary to solve other types of problems when modelling engineering systems. This remains to be investigated.

In Section 4 we will showcase how one can implement the proposed strategy of introducing functions and functional decompositions to explain the survival of engineering systems to component replacement.

3.2. Granularity of the Functional Model

We now turn to the granularity problem:

[2]More precisely, the physical objects whose behaviour executes the function. For simplicity, we omit such details.
[3]It cannot be a standard part-whole relation as shown in [29].
[4]See also [30] for another approach on functional parts.

(pr3) **The granularity problem**. How can we ensure effective management of granularity in engineering system modelling?

We are especially interested in Problem (pr3) when specialized to functional modelling.

Since granularity is an important factor to motivate the choices discussed in this paper, we clarify how it should be here understood in its generality:

d1 Given any binary relation R in some domain of discourse, whose transitive closure is a (possibly strict) partial order, we say that an element x of the domain is at a finer granularity level than the element y, with respect to the relation R, if and only if there exist elements $x_1, x_2, ..., x_n$ such that $R(x, x_1), R(x_1, x_2), ..., R(x_n, y)$.[5] We say that x is at a coarser granularity level than y to mean that y is at a finer granularity level than x.

Since granularity, as defined above, is relative to a partial-order relation and there is no restriction on the number of such relations, a few consequences follow. Given an element, this fixes a granularity level. Given a granularity level, to move to a different (finer or coarser) granularity level, one has to fix the relation of reference. For example, consider a model of a combustion engine. Let us take the engine to be (structurally) composed of three parts: the carburetor (that mixes the fuel with air), the intake manifold (that distributes the air/fuel mix to the cylinders), and the cylinder group. Then, from the point of view of the mereological part-of relation, we have a tree structure, where the root is the engine (as a whole), which has three leaves attached (the carburetor, the manifold, and the cylinder group); and we can move to a finer (coarser) granularity level expanding (collapsing) nodes of this tree structure. Additionally, if a taxonomy[6] of artifact types is used, one could refer to the is-a relation to change granularity level in the taxonomy. Therefore, granularity must be stated relatively to a structure, see Figure 1a for an exemplification.

In the same way, if we have a taxonomy of function-types, together with a relation of aggregation/decomposition of some kind between functions, we can vary the granularity of any functional model along these two different relations. See Figure 1b, which uses the Functional Basis methodology for representing functions, for an exemplification. In this case, moving to a finer granularity level over the aggregation relation means decomposing the current function, while moving to a finer granularity level on the is-a relation is achieved by using more specialized flow and function terms (e.g, from ‘connect’ to ‘mix’).

Granularity is important when modelling engineering systems for, at least, these reasons:

- **Accounting for different viewpoints.** Different operations carried out on a product during the various steps of its lifecycle require different points of view about the product. Those viewpoints may differ also due to the level of granularity they focus on, for example, during the assembly of a machine, a car say, it may be important for the workers to know where each screw goes and what kind of washer is required. On the other hand, during maintenance, it may happen that, if a part of the car is malfunctioning, it is simply replaced as a whole.

[5]The constraint on the transitive closure prevents unwanted scenarios, such as domain elements which are, at the same time, both at coarser and finer granularity level than a second element.

[6]That is, a directed acyclic graph (usually a tree, but sometimes one accounts for multiple inheritance) between classes of entities whose edges represent the subsumption (is-a) relation.

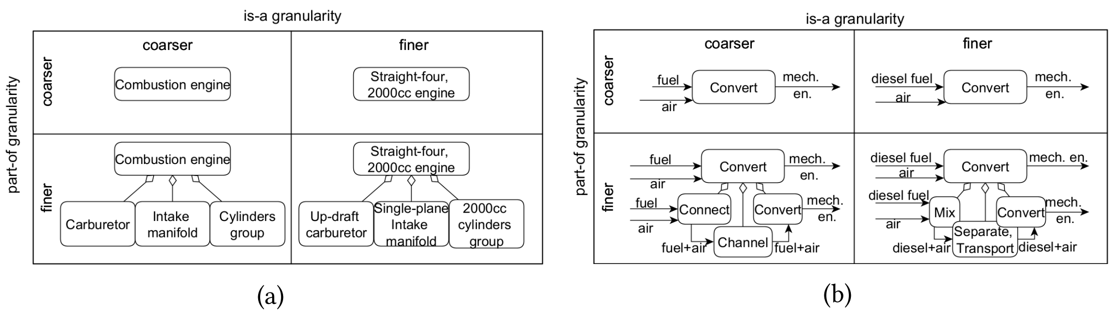

Figure 1: a: A simplified structural model of a combustion engine. Starting from the engine as a whole one can go towards a finer mereological-granularity and find out that the engine is made of three components; going towards a finer subsumption-granularity one can find out additional properties of the engine and its parts.
b: A simplified functional model of a combustion engine using the FB methodology. For simplicity, we refer to the structure in Figure a and assume that each part of a system has a function. Using the FB vocabulary, the function of the engine is converting fuel into mechanical energy, while the function of the carburetor is to connect (more precisely, to mix) fuel and air, and so on.

- **Simplifying the management of complex systems.** Engineering systems can be extremely complex, and when that happens it would be very difficult to manage them using a fixed level of granularity.
- **Reasoning about engineering systems.** Human reasoning about engineering systems often requires jumping between different levels of granularities. For example, the German school of systematic design [2] explicitly describes the design process as the development of a functional hierarchy ('funktionsstruktur') starting from the coarsest level of granularity (the main function the product to be designed should satisfy) and going down to finer-granularity-level functions until one reaches a point that can be readily translated to an assembly of physical items ('embodiment' of the functions). Analogously, a common way for technicians to troubleshoot a malfunctioning system is to produce a chain of parts, each one possibly 'containing' the cause of malfunction and strict subpart of the proceeding part.
- **Allowing scalability of applications.** Suppose that one were to develop an application or a pipeline and test it on a simplified model of, or with a limited dataset from, some system. Later, if the application/pipeline is implemented, the actual model or data inputs may become orders of magnitude bigger than during the testing period. If the behaviour of the application/pipeline concerning granularity was not well thought out, it may not work well.

We are especially interested in the granularity of aggregation-like relations[7] between functions (that is, a functional decomposition), to facilitate the previous points, going towards a solution to Problem (pr3). Moreover, functional modelling is difficult to use in actual engineering

[7]Recall that, generally speaking, one cannot assume that the relation is a part-whole relation [29].

practice, despite its emphasis in the literature; we believe that effective management of granularity is instrumental in facilitating a broader functional modelling adoption in engineering.

Also in this case we argue in favour of a particular style of decomposition, among the few possibilities described in Section 2 (the following considerations can be applied, with some changes, to other functional theories):

- **Functional Basis style of decomposition.** This style of decomposition was briefly described in Section 2, see also Figure 1b. We argue that it has some weak points:
  - It is not (intrinsically) modular: decomposition happens by substituting a box with some input and output flows with a set of boxes with additional flows between them (the original in/out flows are preserved). It may be possible to, say, label some typical decompositions and reuse them, but this is not often discussed in the literature on the Functional Basis, even though some works of the German School of systematic design (e.g. [31]), which the Functional Basis is based on, seem to suggest this.
  - At a finest level of granularity, the decomposition of a system, say an electrical or hydraulic system, can become redundant with the electric/hydraulic schematics already commonly used by engineers. This is intended, since this methodology aims to produce, at the end of the design process, a decomposition that is readily translatable in such schematics. Since we want to cover (at least) maintenance activities, we assume that such schematics already exist, therefore, the decomposition at the finer functional level would be redundant. Moreover, it is generally considered useful to keep the functional and structural representations independent from each other (see e.g. [26]).
  - The taxonomy of the FB vocabulary is organized on three levels only [4]. This, presumably, helps with systematicity, but it imposes only three subsumption-granularity levels (e.g., we cannot distinguish between cooling and heating without expanding the original taxonomy).
- **Functional Representation style of decomposition.** This style is modular by design: we just need to label some CDPs and reuse them. On the other hand, the functions are not organized in a taxonomy, except for the four types determined by Keuneke [32]. Moreover, the view of functions as labels for state transitions/behavioural constraints means that, at finer granularity levels, the decomposition becomes a sort of behavioural diagram, which could even be used to simulate the system as a finite state machine. This may fit some applications, but, arguably, does not supply a language to properly represent a system on the teleological level only.
- **FBRL style of decomposition.** This style is also modular by design: in this case the 'way-of-achievements' can be reused. Additionally, function-classes are organized in a rich taxonomy; even the 'way-of-achievements' are organized in rich taxonomies which implement domain-specific knowledge [33]. Finally, this methodology introduces also teleological relations between functions ('meta-functions') which, arguably, allow the decomposition to stay on a purely teleological level.

In Section 4, we will showcase a simplified application of our interpretation of FBRL-style functional decomposition.

4. How to Put a Possible Approach to Work

The exploitation of the approach we argued for in the previous sections (that is, conceptualizing functions as roles and using a FBRL-style of decomposition) requires further work both at the theoretical and practical levels. Yet, the material introduced in this paper suffices to roughly outline how to apply it in a conceptual model.

First, we need a taxonomy accommodating both physical objects and functions. Figure 2 shows a DOLCE-based [34] putative taxonomy of such a kind, which subsumes functions and 'way-of-achievements' (from now on 'engineering methods') into a concept-category[8]. The categories of functions and methods can then be further specialized into different kinds. Let us assume that these classes are sufficiently characterised ontologically, possibly by adopting a foundational ontology. The classes could be stored, together with relevant info, in an appropriate database (see e.g. [33] for a suitable implementation for engineering methods).

In this approach, a functional decomposition of an engineering system is achieved by alternating the decomposition between engineering methods and functions. This is because, in this approach, a function can be implemented by a method, and a method can raise the problem of implementing some functions of finer granularity. Consider a laser cutting machine that cuts metal tubes using a laser beam emitted from a mobile cutting head. The tubes are held in place and moved by a spindle that rotates and translates. To obtain an acceptable quality of the cut, the workpiece must be kept in the right position very precisely. To do this, a specific subsystem, called *steady rest*, is used. The steady rest secures the tube head by using special jaws moved by pneumatic cylinders. Steady rests are a standard engineering solution in tooling machines (including laser cutting machines), thus they can be considered parts of a well known engineering methodology. In the particular case we are considering, the steady rest is automated (it rotates and performs other things, such as sensing the tube position through a photocell couple), therefore some method must be selected to allow it to perform the needed functions, e.g., *to rotate*. Figure 3 gives an example of how the goal of performing a high quality laser cut is ensured by decomposing the function of a steady rest of the laser cutting machine. The decomposition shows the chosen method based on an *electrical motor* (together with typical auxiliary parts such as a gearbox and a pinion-crown gear coupling). This method can be further divided into a *convert electrical energy into mechanical energy* function, realized by the motor proper, and a *transmit mechanical energy* function, realized by the gears and shafts linked to the motor. The decomposition can go on until the desired granularity level is reached. Note that this functional decomposition manages granularity quite well, in the sense that it is easy to build trees of different depths; and is also modular, as, for example, the motor-gearbox-pinion and the photocell couple methods can be reused many times in the tree.

Finally, as explained in Section 3.1, the replacement of parts can be modelled, by making use of the functional roles as constant 'anchors' for the replaced physical objects. It suffices to

[8]In [34] roles are subsumed into concepts. Here we also subsume functions into concepts, but do not discuss if functions are roles or not.

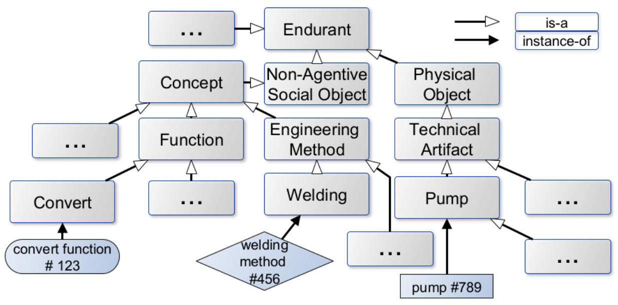

Figure 2: Example of class taxonomy including functions and engineering methods.

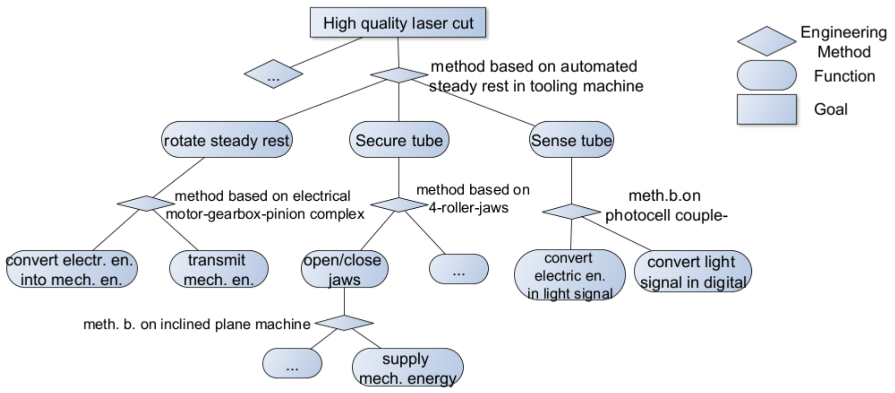
High quality laser cut
...
method based on automated steady rest in tooling machine
Engineering Method
Function
Goal
rotate steady rest
Secure tube
Sense tube
method based on electrical motor-gearbox-pinion complex
method based on 4-roller-jaws
meth.b.on photocell couple-
convert electr. en. into mech. en.
transmit mech. en.
open/close jaws
...
convert electric en. in light signal
convert light signal in digital
meth. b. on inclined plane machine
...
supply mech. energy

Figure 3: Example of functional decomposition. Notice the alternating between functions and methods, which is explained in the text. Additionally, methods should not be confused with the objects required to implement them.

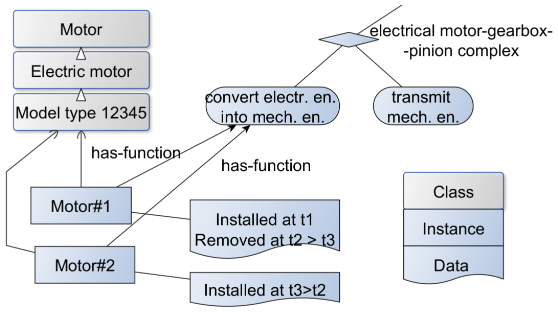

Figure 4: Example of component replacement.

link physical objects with functions, and to, for example, attribute to the physical objects the temporal information about their use (e.g., adding timestamps of installation and removal from the system, or introducing an ‘installed-after’ relation between components). Figure 4 shows such an example: two motors are installed in the same system, at different times, at the same ‘functional location’ (here interpreted simply as some fixed function).

5. Conclusion

The paper described two recurring problems in functional modelling, and used ontological arguments to characterise some of the most common methodologies used in functional modelling with respect to these problems and to show their limitations. The paper suggested that certain characteristics of these methodologies, in particular of the FBRL approach, can be used to successfully manage the two problems. This claim has been supported via an (admittedly brief) example of how the approach could be implemented in conceptual modelling.

Further work is needed to analyze these topics more accurately and to cover additional aspects necessary for the full understanding the proposed methodology including, for example, the modelling of malfunctioning.

Acknowledgments

Francesco Compagno is funded by the company Adige Spa. This work is partially funded by the European project OntoCommons (GA 958371, www.ontocommons.eu).

References

[1] C. Eckert, That which is not form: The practical challenges in using functional concepts in design, Artificial Intelligence for Engineering Design, Analysis and Manufacturing (2013).

[2] G. Pahl, K. Wallace, L. Blessing, G. Pahl (Eds.), Engineering design: a systematic approach, 3rd ed., Springer, London, 2007.

[3] M. Erden et al., A review of function modeling: Approaches and applications, Artificial Intelligence for Engineering Design, Analysis and Manufacturing 22 (2008) 147–169.

[4] J. Hirtz, R. B. Stone, D. A. McAdams, S. Szykman, K. L. Wood, A functional basis for engineering design: Reconciling and evolving previous efforts, Research in Engineering Design 13 (2002) 65–82.

[5] B. Chandrasekaran, A. Goel, Y. Iwasaki, Functional representation as design rationale, Computer 26 (1993) 48–56.

[6] B. Chandrasekaran, J. R. Josephson, Function in Device Representation, Engineering with Computers 16 (2000) 162–177.

[7] M. Sasajima, Y. Kitamura, M. Ikeda, R. Mizoguchi, FBRL: A function and behavior representation language, in: Proceedings of the Fourteenth International Joint Conference on Artificial Intelligence, volume 2, Montreal, Quebec, Canada, 1995.

[8] Y. Kitamura, Y. Koji, R. Mizoguchi, An ontological model of device function: Industrial deployment and lessons learned, Applied Ontology 1 (2006) 237–262.

[9] Y. Umeda, H. Takeda, T. Tomiyama, H. Yoshikawa, Function, behaviour, and structure, Applications of artificial intelligence in engineering V 1 (1990) 177–193.

[10] L. Qian, J. S. Gero, Function–behavior–structure paths and their role in analogy-based design, Artificial Intelligence for Engineering Design, Analysis and Manufacturing (1996).

[11] R. Cummins, Functional Analysis, The Journal of Philosophy 72 (1975) 741–765.

[12] D. Spear et al., Functions in Basic Formal Ontology, Applied Ontology 11 (2016).

[13] H. Herre, B. Heller, P. Burek, R. Hoehndorf, F. Loebe, H. Michalek, General Formal Ontology (GFO) - A Foundational Ontology Integrating Objects and Processes [Version 1.0], Technical Report 8, Institute of Medical Informatics, Statistics and Epidemiology (IMISE), 2006.
[14] S. Borgo, M. Carrara, P. Garbacz, P. E. Vermaas, A formal ontological perspective on the behaviors and functions of technical artifacts, Artificial Intelligence for Engineering Design, Analysis and Manufacturing 23 (2009) 3–21.
[15] R. Arp, S. Barry, Function, role and disposition in Basic Formal Ontology, 2008.
[16] S. Barry et al., Basic Formal Ontology 2.0, 2015.
[17] S. Borgo, M. Carrara, P. Garbacz, P. E. Vermaas, A Formalization of Functions as Operations on Flows, Journal of Computing and Information Science in Engineering 11 (2011) 031007.
[18] P. Garbacz, S. Borgo, M. Carrara, P. E. Vermaas, Two ontology-driven formalisations of functions and their comparison, Journal of Engineering Design 22 (2011) 733–764.
[19] P. Garbacz, Towards a standard taxonomy of artifact functions, Applied Ontology 1 (2005).
[20] P. Burek, R. Hoehndorf, F. Loebe, J. Visagie, H. Herre, J. Kelso, A top-level ontology of functions and its application in the Open Biomedical Ontologies, Bioinformatics 22 (2006).
[21] N. Guarino, Artefactual Systems, Missing Components and Replaceability, in: Artefact Kinds, Springer International Publishing, 2014, pp. 191–206.
[22] M. West, Developing High Quality Data Models, Morgan Kaufmann, Burlington, MA, 2011.
[23] P. E. Vermaas, W. Houkes, Ascribing Functions to Technical Artefacts: A Challenge to Etiological Accounts of Functions, The British Journal for the Philosophy of Science 54 (2003) 261–289.
[24] F. Compagno, S. Borgo, C. Masolo, E. M. Sanfilippo, Comparing Ontological Alternatives to Model Engineering Systems and Components, in: Proceedings of the Joint Ontology Workshops 2021, CEUR, Bolzano, Italy, 2021, p. 12.
[25] J. W. Klüwer, F. Martin-Recuerda, A. Waaler, D. Lupp, M. M. Brandt, S. Grimm, A. Koleva, M. Khan, L. Hella, N. Sandsmark, ISO 15926-14:2020(E), Working Draft, READI, 2020.
[26] ISO/IEC 81346: Industrial Systems, Installations and Equipment and Industrial Products – Structuring Principles and Reference Designation, Technical Report, ISO/IEC, 2009.
[27] R. Mizoguchi, Y. Kitamura, S. Borgo, A unifying definition for artifact and biological functions, Applied Ontology 11 (2016) 129–154.
[28] J. Röhl, L. Jansen, Why functions are not special dispositions: An improved classification of realizables for top-level ontologies, Journal of Biomedical Semantics 5 (2014) 27.
[29] P. E. Vermaas, On the formal impossibility of analysing subfunctions as parts of functions in design methodology, Research in Engineering Design 24 (2013) 19–32.
[30] R. Mizoguchi, S. Borgo, A Preliminary Study of Functional Parts as Roles, in: JOWO, 2017.
[31] K. Roth, Konstruieren mit Konstruktionskatalogen, Springer Berlin Heidelberg, 2000.
[32] A. Keuneke, Device representation-the significance of functional knowledge, IEEE Expert 6 (1991) 22–25.
[33] Y. Kitamura, R. Mizoguchi, Ontology-based description of functional design knowledge and its use in a functional way server, Expert Systems with Applications 24 (2003) 153–166.
[34] S. Borgo et al., Dolce: A descriptive ontology for linguistic and cognitive engineering, Applied Ontology 17 (2022) 45–69.