Proceedings of the ASME 2007 International Design Engineering Technical Conferences & Computers and Information in Engineering Conference
IDETC/CIE 2007
September 4-7, 2007, Las Vegas, Nevada, USA

DETC2007-34232

FUNCTIONAL DECOMPOSITION IN ENGINEERING: A SURVEY

Dingmar van Eck*
Department of Philosophy
Delft University of Technology
Jaffalaan 5, 2628 BX Delft
The Netherlands

Daniel A. McAdams
Department of Mechanical & Aerospace Engineering
The University of Missouri-Rolla
Rolla, MO 65409, USA

Pieter E. Vermaas
Department of Philosophy
Delft University of Technology
Jaffalaan 5, 2628 BX Delft
The Netherlands

ABSTRACT

Functional reasoning is regarded as an important asset to the engineering designers' conceptual toolkit. Yet despite the value of functional reasoning for engineering design, a consensus view is lacking and several distinct proposals have been formulated. In this paper some of the main models for functional reasoning that are currently in use or discussed in engineering are surveyed and some of their differences clarified. The models included the Functional Basis approach by Stone and Wood [1], the Function Behavior State approach by Umeda et al. [2, 3, 4], and the Functional Reasoning approach of Chakrabarti and Bligh [5, 6]. This paper explicates differences between these approaches relating to: (1) representations of function and how they are influenced by design aims and form solutions, and (2) functional decomposition strategies, taken as the reasoning from overall artifact functions to sub-functions, and how these decomposition strategies are influenced by the use of existing engineering design knowledge bases.

1 INTRODUCTION

Functional reasoning is increasingly regarded as an important technique in engineering [7, 8]. For instance, in engineering designing of innovative products functional reasoning is required in the initial conceptual phase of the design process to fix the functional structure of a product relatively independently of specific design solutions, and in engineering descriptions and redesigning of existing products, functional reasoning complements physical descriptions of the products indicating the roles and uses of products and their components. Moreover, functional reasoning is increasingly regarded as a common technique, which is shared in engineering communication about functional structures of products that are (collaboratively) designed or redesigned, and in the archiving and use of functional structures of existing products in knowledge bases.

This importance of functional reasoning in engineering warrants an ongoing effort at formalization to arrive at uniform and commonly shared methods for the modeling of functions, to fix the relations that hold between functions, and to define algorithms that support reasoning about functions. Such formalization would not only facilitate the above-mentioned uses of functional reasoning, but also allow for the next step of automating functional reasoning in computer tools, ranging from CAD-CAM systems to engineering knowledge bases.

Yet, despite the broadly accepted value of a common and uniform language for functional reasoning, the current situation in the engineering sciences is rather one of plurality. Just as that there is no consensus in engineering about the meaning of the term function itself [9, 10, 11], there is currently a series of functional reasoning models being proposed and developed that are distinct rather than convergent.

This plurality has – apparently – not hampered engineering and designing so far. Yet, with the increasing use of computer

* Author of correspondence, Phone: + 31 15 278-5417, Fax: + 31 15 278-6233, Email: D.vanEck@tudelft.nl.

tools such as CAD-CAM systems and engineering knowledge bases, and with the increasing effort to couple these systems by communication or integration, it may be envisaged that this lack of consensus may play up at some point in the near future. If, for instance, the current models for functional reasoning are developed successfully in competition with one another, and then incorporated in separate engineering computer systems, one is bound to end up with design tools and knowledge bases that are difficult to integrate. The source of this problem will then be identified as consisting of the different models of functional reasoning underlying these computer systems, which will then become topic of analysis. Anticipating these problems, and – possibly somewhat ideally – trying to prevent them, we here set out in this paper to give this analysis immediately by surveying a number of the main models of functional reasoning, as currently discussed in the engineering literature.

We focus in this survey on conceptual differences between three models of functional reasoning, specifically on how the term function itself is understood in these models, on how functional decomposition taken as the reasoning from overall functions to sub-functions is captured, and on how this decomposition of functions is related to engineering knowledge about design solutions to functions. These models are the *Functional Basis* approach by Robert Stone and Kristin Wood [1], the *Function Behavior State* approach by Yasushi Umeda et al. [2, 3, 4], and the *Functional Reasoning* approach by Amaresh Chakrabarti and Thomas Bligh [5, 6]. These three models were chosen because they describe functional reasoning explicitly but reveal already on first inspection relevant differences. First, in the Function Behavior State approach design reasoning from functions to physical structure takes place via the intermediate notion of behavior, whereas in the other two approaches this reasoning is done without an intermediate concept. Second, the Functional Basis approach advances functional reasoning that is independent of existing design solutions to functions, whereas in the other two functional reasoning explicitly depends on such solutions.

The sections 2, 3, and 4 are used for this presentation. Then, in section 5, we briefly compare the key elements of these models and in section 6 we end with conclusions.

2 THE FUNCTIONAL BASIS APPROACH

The Functional Basis approach by Robert Stone and Kristin Wood is an approach to that aims at creating a common and consistent functional design language, dubbed a functional basis, which allows designers to describe overall product functions in terms of interconnected sub-functions [1].

The Functional Basis approach is focused on especially the electromechanical and mechanical domain and is presented as supporting the archiving, comparison, and communication of functional descriptions of existing products, as well as the engineering designing of new products. Archiving, comparison, and communication are assisted since the sub-functions into which overall product functions are decomposed, are described in a common universal language. Designing of new products is supported since functional modeling allows designers to make critical design decisions about the product's architecture in the early conceptual stage of designing at which only functional descriptions are considered.

Functions and sub-functions in the Functional Basis approach are captured in terms of operations on flows of materials, energies, and signals. This operation on flows description of functions is reminiscent of the work on functions by Pahl and Beitz [12].

2.1 Functions as Operations on Flows

In the Functional Basis approach an overall product function refers to a general input/output relationship of a products' overall task, which is described in a verb-object form and represented by a black-boxed operation on flows of materials, energies, and signals. A sub-function is also described in a verb-object form but represented by a well-defined basic operation on well-defined basic flows of materials, energies, and signals. The black-boxed operations on general flows representing product functions are derived from customer needs, and the basic operations and basic flows representing sub-functions are laid down in common and limited libraries that span the functional design space. These libraries are called a *functional basis* (the current libraries have been fixed in [8] by integrating the libraries proposed in [1] with similar libraries developed at the US National Institute of Standards and Technology [13]).

To give an idea on how overall product functions and connected sub-functions are represented in the Functional Basis approach, consider the example of a power screwdriver (adapted from [1]); the overall product function of the screwdriver is represented (in verb-object form) as the operation "loosen/tighten screws", and has input and output flows of energies, materials, and signals. This overall product function representation thus black boxes the internal flows of the power screwdriver that transform the input flows into output flows. Focusing on the energy flow, the input side consists of flows of electricity, human force, relative rotation, and weight, and the output flows consist of torque, heat, noise, human force, and weight. Extracting the electricity part of the (internal) energy flow, the representation of the detailed (temporally ordered) sub-functions operating on the incoming electricity flow are represented as "store electricity", "supply electricity", "transmit electricity", "actuate electricity", and "regulate electricity", respectively. These sub-functions are thus represented as basic operations on basic flows and transform an input flow step-by-step into an output flow.

Table 1. Primary basic operations and flows in the functional basis [8].

Primary basic operations	Branch, Channel, Connect, Control Magnitude, Convert, Provision, Signal, Support
Primary basic flows	Material, Signal, Energy

2.2 Developing Sub-Function Chains

The Functional Basis approach of Stone and Wood allows, as said, to model overall product functions as sets of connected sub-functions. Their functional modeling framework provides the means to decompose overall product functions into what they call 'small, easily solvable sub-functions', i.e., sub-functions for which solutions exist such that 'the [structural] form of the device [product to be designed] follows from the assembly of all sub-function solutions' [1].

These sub-function assemblies, i.e., function structures, result from the following decomposition steps or tasks:

The first task is to arrive at an overall product function of a product to be designed, described in a verb-object form and represented by a black-boxed operation on flows of materials, energies, and signals. This black-boxed operation originates from customer needs and may initially be quite general, and is later on in the design process refined. The verb-object description of the product function and this (black-boxed) operation-on-flows representation are related in the sense that the verb corresponds with the operation and the object corresponds with (parts of) the flows.

The second task is to define for each input flow a chain of sub-functions that transform that flow step-by-step into an output flow. These sub-functions are also described in verb-object forms and represented by operations on flows. But now the verbs are to be chosen from the functional basis. The sub-functions part of the different chains must be ordered in time with respect to one another.

The third task is then to integrate these temporally ordered chains of sub-functions by connecting them, thereby arriving at the functional model. In this three-step process the designer thus has to analyze the input flows of the overall function in terms of basic flows from the library of basic flows, and come up with a series of operations from the library of basic operations that sequentially and/or in parallel transform the input flows step-by-step into the output flows. This transition from input to output is expressed by Stone and Wood thus: "think of each operation on the flow from entrance until exit of the product (or transformation to another flow) and express it as a sub-function in verb-object form" [1].

The Stone-Wood approach allows functional models to be developed with a high level of detail. It is argued that the functional basis defines a comprehensive set of basic functions, able to fully span the mechanical (and electromechanical) design space, and that these basic functions have a level of detail, which allows them to be "easily solvable" [1].

2.3 Design Repositories and the Concept Generator

Having discussed the function definitions and functional decomposition strategy of the Functional Basis approach, we discuss in this section a design tool called the Concept Generator that can be taken as a natural extension of the Functional Basis approach [14]. The Concept Generator is an automated, mathematically based design tool that aids in the development of functional models by retrieving design solutions to sub-functions from a repository archiving functional models of products. It thus is a tool that aids in and implements a main objective of functional decomposition, i.e., the generation of functional design solutions. The other functional decomposition approaches surveyed in this paper also employ such archived design knowledge bases in generating functional models, and we therefore include the Concept Generator in our survey in order to be able to discuss in section 5 the different ways in which knowledge bases assist in developing functional models.

Table 2. Function definitions, decomposition strategy, and design knowledge in the Functional Basis approach.

Function	General input/output relationship of a product having the purpose of performing an overall task, represented by a black box operation on flows of material, energy, and signal, determined by costumer needs
Sub-function	Part of a product's overall task, represented by a basic operation on basic flows of material, energy, or signal, as defined by the functional basis libraries of basic operations and basic flows
Decomposition strategy	Analysis of the black-box operation on flows corresponding to the overall product function in terms of connected basic operations on basic flows corresponding to the sub-functions
Design knowledge base	Archived functional models of products and archived components that counts as design solutions to sub-functions

The Concept Generator can be found at a site created at the Design Engineering Lab of the University of Missouri-Rolla, which hosts also a web-based repository that contains functional models of up to 102 products and their components, and that stores the design solutions – the physical structures that can perform these sub-functions – for the sub-functions part of these models [15]. The Concept Generator is now aimed at creating new functional models for any overall product function that is fed into it, and at generating design solutions for these overall product functions on the basis of the design solutions of their sub-functions that are already stored in the repository.

In the first step of the algorithm, the Concept Generator translates the functional model of the overall product function (only models consisting of single chains of sub-functions are considered in [14]) into information about the sub-functions and their adjacency. Secondly, the Concept Generator collects for each individual sub-function in the chain design solutions consisting of components that are stored in the repository as having that specific sub-function. In the third step, all design solutions for the product as a whole are generated by describing, on the basis of the information gathered in the first two steps, all theoretically possible component chains that solve

the overall product function. Fourthly, additional information is collected from the repository on which sets of components have been actually combined in existing products, and in that sense can be taken as sets of compatible components. Finally, in a fifth step, this information about the compatibility of components is used to prune the set of theoretically possible component chains to a set of feasible component chains that solve the overall product function. In this final step a ranking is also added to these feasible chains, to "bubble the most promising solutions to the top". The algorithm puts constraints on the functional modeling of overall product functions: the algorithm selects only those functional models that consists of sub-functions for which there are components available in the design repository that have these sub-functions, and the algorithm selects only those models that combine sub-functions that are actually combined in functional models of products in the repository.

To be sure, there is more to be said on the algorithm of the Concept Generator and on the way it is implemented in computer systems. For the purposes of this paper, i.e., comparing the conceptual differences between models of functional reasoning, the above description suffices (but see [14, 15] for more details).

3 THE FUNCTION BEHAVIOR STATE APPROACH

Like the Functional Basis approach, the Function Behavior State (FBS) approach developed by Yasushi Umeda et al. [2, 3, 4] aims to support and facilitate functional design. In particular, the FBS approach is reported to support the synthetic phase of functional design, described by its developers as the process by which functional descriptions representing user intentions are transformed into structural descriptions of products-to-be [2]. To support such synthetic design tasks, the FBS approach employs a computer-based design tool, called the FBS modeler, which generates functional models that represent mappings amongst functions, behaviors, and product structures. Structures are referred to as states in the FBS approach and the state of a design object is represented by entities, their attributes, and relations between entities [2]. The FBS modeler thus is a design tool supporting "the embodiment process of a function" [4], in which the concept of behavior is used as an intermediate step in transforming functions into states or structures.

3.1 Functions as Intended Behaviors

In the FBS approach functions are equated with intended behaviors and described thus: "a description of behavior abstracted by human through recognition of the behavior in order to utilize it" [16] and elsewhere as "a function is an association between the designer's intention and a behavior that realizes the function" [3]. A function is represented by an association of two concepts; a verb-object pair that represents designer intentions, expressed in the form of "to do something" [2, 4], and behavior(s) that can instantiate the function. Behaviors in the FBS approach are defined as "sequential state transitions along time" [2].

Functions are archived in so-called function prototypes. Function prototypes are designer knowledge bases which represent and store typical patterns of functions that often appear in a certain design domain [4]. Instead of defining functional primitives, in the FBS approach, functions are defined by developing and archiving various function prototypes from existing product designs, basing representations of functions on them. Function prototypes represent, amongst others, functions and networks of sub-functions into which the function can be decomposed, which the FBS modeler employs in developing functional models of design objects (see section 3.3).

An example (adapted from [2, 4]) of function, behavior, and state representations in the FBS modeler is, for instance, the function "to charge drum" in the design of a photocopier. In the FBS modeler, this function is decomposed into the sub-functions "to rotate drum" and "to discharge voltage to drum", of which the latter sub-function is achieved by the behaviors "electrical charging" and "electrical discharging", occurring on the entities (states) "drum" and "discharger", respectively.

3.2 Developing Function-Behavior-State Mappings

Functional design in the FBS approach, employing the FBS modeler, consists of six steps that involve two distinct types and phases of functional decomposition in which two distinct knowledge bases are used [2, 3]: First, the designer specifies a required function by selecting an appropriate function prototype [2]. This specification and selection is somewhat unclear in the FBS approach, but the authors seem to suggest that choosing a function prototype fixes what the required function will be, instead of an appropriate function prototype being selected based on the (prior) formulation of a required function (see section 3.1). Second, the required function is then, by applying decomposition knowledge stored in the selected function prototype, decomposed into sub-functions, a procedure coined "task decomposition" [2]. Third, these sub-functions are then mapped onto design solutions by instantiating appropriate *physical features* selected by the designer. Physical features, also archived in the selected function prototypes, refer to sets of components, their relations, and physical phenomena that together instantiate specific behavior(s) and thus particular function(s) [2].

Fourth, after instantiation of physical features it may be the case that some of them cannot occur because not all the requisite physical conditions to instantiate the physical phenomena are met. For instance, for a particular physical phenomenon to occur, additional components might be needed, and connected to the ones already selected via physical feature instantiation. Resultantly, not all task-decomposed functions can be mapped onto design solutions. If this occurs, a subsystem of the FBS modeler, coined Qualitative Process Abduction System (QPAS), reasons out additional physical features to realize these conditions, thereby realizing the embodiment of the sub-functions that remained un-instantiated after the task decomposition. This procedure is dubbed "causal

decomposition". QPAS is a knowledge-based reasoning system that generates design solution candidates by using physical phenomenon and physical feature knowledge bases [17]. It takes as input a particular (series of) state transition(s), including relevant entities and physical conditions, and then derives candidate physical phenomena that can achieve the state transition(s). Subsequently, QPAS derives physical features that include the physical phenomenon, after which the designer can select an appropriate one, thus instantiating the physical phenomenon [17] (see also [18] for additional details on QPAS).

Then, after physical feature selection and instantiation, the FBS modeler again employs the function prototypes knowledge base and reasons out and selects a function prototype that has the added physical feature in its function-behavior relation and thus explains the function(s) of the added physical feature. The function(s) is (are) then incorporated in the FBS model, and connected as a cause to the initially un-instantiated sub-function(s) of the task decomposition.

In the next fifth step, the designer constructs a behavioral network by connecting the instantiated physical features, completing the functional hierarchy. Then, in the sixth step, a behavior simulation is run that evaluates the FBS model. If the simulation is unsatisfactory the designer can make adjustments in either the functional hierarchy or in the behavioral network.

This six-step functional decomposition strategy thus involves two distinct decomposition phases and types; a task decomposition in the first phase (step 2) based on knowledge stored in function prototypes, and a causal decomposition in the second phase (step 4) based on knowledge derived from QPAS [2]. The difference between the decomposition types seems to be one of detail; in task decomposition, the (sub) functions reflect designer intentions ("to do something") and their instantiating behaviors are not causally dependent on one another and cannot be derived from physical knowledge [2]. By contrast, (sub) functions in causal decompositions are physically derivable and their instantiating behaviors are causally related [2]. Umeda et al. [2] explain this difference thus: "the task-decomposed functional hierarchy can be considered as a description of the designer's intention, the extracted functions by the causal decomposition can be considered as additional functions for explaining the mechanism to realize the designer's intention" [2]. Their reference to mechanisms realizing a functional hierarchy suggests that functions in casual decomposition are more specific than functions in task decomposition.

3.3 Function Prototypes

The abovementioned function prototypes refer to designer knowledge bases that archive and represent typical patterns of functions for a certain design domain [4]. The FBS strategy for defining functions, instead of an a priori method, consists in collecting various function prototypes from existing design results, and then to base representations of function on these prototypes [2]. The schemes within function prototypes represent task functions that include a symbol for the designer's intention, a network of sub-functions into which the task function can be decomposed, as well as function-behavior relations that are represented in the form of physical features, thus specifying candidate embodiments of the (sub) functions. Function prototypes thus contain archived function-behavior-state relations or mappings, derived from previous design results. Physical features are viewed as "building blocks" of functions, and consist of entities, relations between entities, and physical phenomena occurring on them. In the FBS modeler these elements of physical features are dubbed behavioral nodes, which together comprise a behavioral network [2]. As discussed in the last section, the FBS approach relies on these function prototypes in specifying required functions and in the first phase (steps 1-3) of its functional decomposition scheme where function-behavior-state-mappings are developed. Next to these function prototypes, in the second phase of the FBS decomposition strategy (steps 4-6), the FBS modeler employs QPAS that uses physical phenomenon and physical feature knowledge bases to realize un-instantiated functions from the first phase by finding out "appropriate mechanisms for realizing task-decomposed functions with physical knowledge" [2].

Table 3. Function definitions, decomposition strategy, and design knowledge in the FBS approach.

Function	Intended behavior, expressed as a combination of designer intentions and behavior(s) that can exhibit the function
Sub-function	No separate definition given: sub-functions are part of an overall function, derived from either causal decomposition or task decomposition
Decomposition strategy	Function-state mapping via behaviors and physical features. Two distinct decomposition phases and types: first, task decomposition and resulting function-state mapping; second, causal decomposition and resulting function-state mapping of task-decomposed functions that could not be mapped onto states in the first phase
Design knowledge base	Archived function-behavior-state mappings

4 THE CHAKRABARTI-BLIGH APPROACH

The Functional Reasoning approach developed by Amaresh Chakrabarti and Thomas Bligh [5, 6] is yet another framework for functional reasoning in the conceptual mechanical design space, in particular the domain of mechanical transmission design [5]. It aims to facilitate computational approaches to design and takes a prescriptive stance towards designing. Chakrabarti and Bligh [6] identify three requirements of an ideal functional reasoning approach: it should support design of any nature, from routine to innovative, it should support the

synthesis of design solutions at a given level of design, and it should support the elaboration of solution concepts through levels of detail [6]. They argue in [6] that these aims can only be met when function based reasoning approaches employ a known set of design solutions (structures) in its reasoning schemes. These solutions or structures are to be represented in terms of the functions they provide or require (to operate properly). In the Chakrabarti-Bligh approach, the aim of functional reasoning is taken to provide "physical descriptions of designs, sufficient for their implementation, which would provide the intended functions of the problem" [6]. This approach consists of a piecemeal approach in which function-component mappings are integrated from the outset (and step by step) in a functional reasoning process until the solution space (of known structures) has been searched completely. It thus tackles the design problem, i.e., the realization of the overall product function, in parts.

4.1 Function as Action or Effect

In the Chakrabarti-Bligh approach, function is understood as a description of the action or effect required by a design problem or supplied by a solution [6]. In earlier work of Chakrabarti and Bligh, function is exclusively tied to an artifact's structure or solution and expressed as a "description of the action or effect (intended to be) produced by an object, i.e., what it (is intended to do or) does" [5]. So, functional representations in this approach describe what a structure does, or is supposed to do [6], i.e., objects are described in terms of their known functions [5].

Functions are, moreover, defined in terms of mathematical expressions [5, 6] that express transformations between a set of input characteristics and a set of output characteristics. For instance, in [5] the example of a bicycle drive is presented where a particular structure of this device has the function "to transform input torque into output torque" and a connected structure has the function "to transform input torque into output force". These functions of physical entities are, moreover, classified into three distinct types: structures that have as their primary role the transmitting or transforming of force and energy (like in the example above); structures that have couple or connect functions; and structures that take unwanted forces away [5]. These function types are defined for the domain of mechanical transmission design for which they take input-output motion transformations as functional primitives. Such a classification of functions into distinct types has also been proposed by other design methodologies such as Pahl and Beitz's [12] and Hubka and Eder's [19]. Pahl and Beitz, for instance, distinguish main functions from auxiliary functions.

4.2 Recursive Problem Redefinition

Chakrabarti and Bligh's model for functional reasoning consists of several steps and presupposes the existence of a given set of design solutions, and a design problem defined in terms of a function or a set of functions. Moreover it is required that these design solutions can solve the design problem. The steps included in their Functional Reasoning approach to arrive at a design solution for an overall design problem are [6]: First, the overall problem is described in terms of the intended function(s) of the physical design to be developed. A part of this problem, the sub-function, is then chosen for synthesis, i.e., for function-structure mapping. Then, second, from a known set of structures, alternative solutions are chosen that can instantiate the sub-function. Third, the first of these alternative solutions is chosen and its (sub) function is evaluated with respect to the overall function or design problem. By incorporating the chosen part-solution, the overall function is revised and narrowed down; the number of sub-functions for which solutions have to be found is decreased. Fourthly, another sub-function of the now revised overall function is selected for synthesis, i.e., structure-function mapping. This step-by-step recursive procedure continues until all the sub-functions of the overall function are solved, i.e., all realizing structures are found. These steps thus solve a problem part (sub-function) of the overall design problem (overall function) at a time, thereby reducing the overall problem space, until the overall function is completely solved and implementing structures are identified for each sub-function. This four-step procedure leads to the generation of a single solution structure, being an aggregate of the partial solutions chosen. (The Chakrabarti-Bligh approach is prescriptive in nature; the steps above are abstractly defined and the authors do not present rules of thumb on how to select partial solutions for an overall design problem).

These steps lead to the development of a single solution structure. Then, to find alternative solution structures, the functional reasoning strategy continues by going back one step and choosing another partial solution resulting in another revision of the overall function. Then one repeats this step by going back yet another step and choosing again another partial solution which solves a part of the revised overall function, thereby revising it again. These steps continue until the complete solution space has been searched, i.e., all possible partial solutions for sub-functions have been identified and thus all possible configurations of solution aggregates (component configurations) have been identified that can solve the overall design problem or function [6].

To summarize, the functional reasoning scheme of Chakrabarti and Bligh amounts to a piecemeal, sub-function to component mapping approach incorporating these function-component mappings at each successive stage of the reasoning process (and then back again) until the solution space (of known structures) has been searched completely. It thus thereby tackles the problem, i.e., the realization of the overall device function, in parts.

4.3 Known Physical Solutions

The inclusion of known solutions in the Functional Reasoning approach of Chakrabarti and Bligh is derived from the idea that function structures cannot be developed usefully in a solution-neutral way [6]; without known solutions for function realization there is no guarantee that one moves from problem

state towards solution space. The lynchpin of this argument is what they refer to as the "problem of partitioning" [6]: the same component may support multiple sub-functions and thus there is not always a one-to-one mapping between components and sub-functions. A set of sub-functions *together* can have one solution and this many-one mapping would remain undetectable in a solution-neutral reasoning scheme, the argument goes. Moreover, a given function vocabulary is likely to be incomplete and therefore, when it would be developed in a solution-neutral way, the possibility exists that it cannot express the functions of all structures. The other way around, by ignoring solution concepts from the outset, the possibility exists that function structures are developed which cannot be solved in terms of known solutions or structures. To circumvent these possible problems, Chakrabarti and Bligh [6] incorporate function-structure relations, based on known mappings, from the outset; structure-function couplings are incorporated from start to finish in their recursive problem reformulation approach.

The approach of Chakrabarti-Bligh is prescriptive in nature and suggests how functional reasoning schemes ought to be developed and used. Their approach, at least as laid down in [5] and [6], does not employ existing design knowledge bases, but instead prescribes that such knowledge bases are to be used in functional reasoning schemes.

Table 4. Function definitions, decomposition strategy, and design knowledge in the Chakrabarti-Bligh approach.

Function	The intended action or effect of a physical object, expressed in terms of mathematical input-output transformations
Sub-function	Function of a component of a design object, expressed in terms of mathematical input-output transformations
Decomposition strategy	Recursive function-structure mapping, incorporating partial function-structure solutions in each successive stage of the reasoning process
Design knowledge base	Known function-structure mappings

5 COMPARISON

Having discussed all three approaches, in this section they will be compared. We will focus in particular on how the term function itself is understood in these approaches, how functional decomposition taken as the reasoning from overall functions to sub-functions is captured, and on how this decomposition of functions is related to engineering knowledge about design solutions to functions.

5.1 Diverging Function Definitions: Form-Independency

Whereas the three approaches take a product function to be intended or required entities that are derived from user or customer demands (although the latter is not made explicit in the Chakrabarti-Bligh approach [5, 6], we assume that their scheme underwrites this), they nevertheless put forward somewhat different definitions of functions.

Table 5. Function in the Functional Basis, FBS, and Chakrabarti-Bligh approaches.

Functional Basis	General input/output relationship of a product having the purpose of performing an overall task, represented by a black box operation on flows of material, energy, and signal determined by customer needs
FBS	Intended behavior, expressed as a combination of designer intentions and behavior(s) that can exhibit the function
Chakrabarti-Bligh	The intended action or effect of a physical object, expressed in terms of mathematical input-output transformations

Table 6. Sub-function in the Functional Basis, FBS, and Chakrabarti-Bligh approaches.

Functional Basis	Part of a product's overall task, represented by a basic operation on basic flows of material, energy, or signal, as defined by the functional basis libraries of basic operations and basic flows
FBS	No separate definition given: sub-functions are part of an overall function, derived from either causal decomposition or task decomposition
Chakrabarti-Bligh	Function of a component of a design object, expressed in terms of mathematical input-output transformations

A main difference between these approaches concerns the manner in which they implement forms or objects in their definitions of function, as will become clear below. This difference is likely related to the different aims that the respective approaches associate with function-based designing. Whereas a functional model in the Functional Basis approach is regarded as a "form-independent blueprint" [1] of an artifact, the main task of functional design is in the FBS approach regarded as "the embodiment process of required functions" [4]. In line with this, Chakrabarti and Bligh [6] argue that known structure-function solutions are to be incorporated from the outset in functional reasoning schemes, if the solution space is to be searched exhaustively. More schematically, the three approaches take functions, respectively, as *form-independent operations on flows*, as *intended behaviors of forms or objects*, and as *intended transformations of forms or objects*.

The Functional Basis approach defines a function as "a description of an operation to be performed by a device or artifact" [1] and, in accordance with the aim to deliver form independent functional models, its descriptions of basic operations on basic flows do not include reference to form solutions. Functions in this approach thus refer to form-independent operations. For instance, in the power screwdriver example given in section 2.1 the sub-functions operating on the electricity flow are all defined independently from any component solutions that can implement them.

By contrast, in the FBS approach form assumptions are incorporated in its representation of function. Consider, for instance, the example provided in section 3.1 where the overall function "to charge drum" is decomposed, in the FBS modeler, into the sub-functions "to rotate drum" and "to discharge voltage to drum". This example shows that, in the FBS approach, functional descriptions of overall functions and sub-functions already include form solutions. This is in line with the FBS design aim to arrive at the embodiment level of functions.

Moreover, function representations in the FBS approach do not have the level of detail or granularity that the basic functions of the functional basis have, and are not expressed in an input-output operation-on-flow type style, but instead represented in terms of the above generic functional descriptions.

In the Functional Reasoning approach of Chakrabarti and Bligh, functional representations describe what structures do or are supposed to do, and thus refer to the functions of objects. These are furthermore expressed as transformations between a set of input characteristics and a set of output characteristics. Recall the bicycle drive example of section 4.1, in which structures of the bicycle drive are ascribed a certain function, such as "to transform input torque into output force" [5]. In one way, this resembles the Functional Basis notion of functions in the sense that function representations in the Chakrabarti-Bligh approach [5, 6] are, like functions in the Functional Basis approach, defined in terms of transformations/operations. On the other hand, there is a difference; whereas the Functional Basis approach defines functions in a form-independent manner, Chakrabarti and Bligh speak explicitly of the functions of physical parts or objects. This definition of functions (likely) relates to the prescriptive aims that are associated with functional designing, i.e., the development of meaningful function structures is contingent on using known structure-function couplings, i.e., object functions (see section 4.3).

The object function definition of the Chakrabarti-Bligh approach is similar to the FBS approach in the sense that in the latter function definitions also include (assumptions about) forms. There are however also some differences: as said, the FBS framework does not express functions in terms of transformations and, moreover, its functional descriptions are more broadly defined than object functions in the Chakrabarti-Bligh approach. For instance, the object function "a shaft transmits torque" [5] is more specific than the functional description "to move the table with a motor fast" [2]. Whereas the former defines the function of an object (shaft), the latter description explicates that an object (motor) will be used to move another object (table), but does not stipulate a precise object function.

So, summing up, these three approaches differ somewhat in how they represent functions, to wit: in the Functional Basis approach functions are form-independently defined, in the FBS approach representations of function include object or form solutions, and in the approach of Chakrabarti and Bligh functions are represented as the functions of physical objects. In the next section we suggest that these different definitions have an effect on the proposed strategies regarding the development of function structures.

5.2 Decomposition Strategies Decomposed: Using Design Knowledge Bases in Different Ways

The inclusion or exclusion of form assumptions in functional definitions seems to lead to different steps taken in the reasoning from overall product functions to sub-function assemblies.

Table 7. Decomposition strategy in the Functional Basis, FBS, and Chakrabarti-Bligh approaches.

Functional Basis	Analysis of the black-box operation on flows corresponding to the overall product function in terms of connected basic operations on basic flows corresponding to the sub-functions
FBS	Function-state mapping via behaviors and physical features. Two distinct decomposition phases and types: first, task decomposition and resulting function-state mapping; second, causal decomposition and resulting function-state mapping of task-decomposed functions that could not be mapped onto states in the first phase
Chakrabarti-Bligh	Recursive function-structure mapping, incorporating partial function-structure solutions in each successive stage of the reasoning process

Table 8 Functional knowledge bases in the Functional Basis, FBS, and Chakrabarti-Bligh approaches.

Functional Basis	Archived functional models of products and archived components that counts as design solutions to sub-functions
FBS	Archived function-behavior-state mappings
Chakrabarti-Bligh	Known function-structure mappings. The approach is prescriptive and does not employ existing knowledge bases

The Functional Basis approach of functional modeling translates overall product functions in a three-step sequence into functional models consisting of highly specific and form-

independent sub-functions. These functional models consist of temporally organized and interconnected basic operations on basic flows, excluding references to implementation details. After these functional models have been developed, the Concept Generator produces design solutions for the individual sub-functions on the basis of existing design solutions of sub-functions that are stored in a repository. Use of this repository in the Functional Basis approach does not commence in the early three-step phase of developing functional models, at least not in innovative designing (and not in [1]). The repository is essentially a knowledge base which stores known functional models and sub-function to component mappings from actual products, but is not a necessary add-on in the development of functional models, i.e., function structures of products-to-be. These functional models are developed, using the functional basis libraries and without employing design repositories.

In contrast, in the FBS approach the use of function prototypes, from which function definitions are derived, is an integral and necessary part of functional designing. From the start, and in the different decomposition phases, the FBS approach employs archived function relations and function-structure couplings in its functional decomposition scheme. Consider, for instance, the following example of functional decomposition knowledge extracted from a function prototype [2]: the function “to move a table fast and precisely” is decomposed into the sub-functions “to move the table with a motor fast” and “to stop the table precisely”. This example shows that, in the FBS approach, functional descriptions of overall functions and sub-functions already include form solutions and that the possible functional decompositions in the FBS approach are constrained by archived solutions regarding the embodiment of overall functions and sub-functions, i.e., in the above example that a motor should be used to move the table fast. This thus shows that, in the FBS approach, function definitions and the ways in which an overall function is decomposed are constrained by the use of design knowledge bases.

The Chakrabarti-Bligh approach of functional reasoning is similar to the FBS approach (and different from the Functional Basis approach) in the sense that it also, from the outset, employs design knowledge bases, storing structure-function relations, in its functional reasoning scheme. A difference with the FBS approach is that in the Chakrabarti and Bligh approach, alternative function-structure solutions must be incorporated in the functional reasoning steps in order to search the complete design solution space. It must be said that this suggestion is prescriptive in nature and not descriptive; in contrast with the Functional Basis approach and the FBS approach, the approach of Chakrabarti and Bligh suggests how function based reasoning ought to proceed and does not employ existing design knowledge databases [6].

To sum up, whether functions are form-independently defined or not seems, in the surveyed approaches at least, to be related to the manner in which design knowledge bases are employed in developing functional decomposition schemes. Whereas in the Functional Basis approach the use of knowledge bases is not primary in developing functional models, these knowledge bases are an integral part in the FBS approach and the Chakrabarti-Bligh approach. As a result, form solutions are interwoven in the decomposition strategies of the latter two approaches, whereas in the former approach they are not; function-component solutions are stored in the design repository, but are not employed in functional model development.

6 CONCLUSIONS AND OUTLOOK

In this paper we have surveyed the Functional Basis approach of Stone and Wood [1], the FBS approach of Umeda et al. [2, 3, 4], and the Functional Reasoning approach of Chakrabarti and Bligh [5, 6]. The focus in this survey was on conceptual differences between these approaches, specifically on how the term function itself is understood, on how functional decomposition taken as the reasoning from overall functions to sub-functions is captured, and on how this decomposition of functions is related to engineering knowledge about design solutions to functions. It is argued that the Functional Basis approach defines functions in a form-independent fashion, that in the FBS approach representations of function include object or form solutions, and that in the approach of Chakrabarti and Bligh functions are represented as the functions of physical objects. It has further been argued that these function definitions are related to the manner in which the respective approaches employ design knowledge bases in developing functional decomposition schemes. Whereas in the Functional Basis approach the use of knowledge bases is not primary in developing functional models, these knowledge bases are an integral part in the FBS approach and the Chakrabarti-Bligh approach. As a result, form solutions are interwoven in the decomposition strategies of the latter two approaches, whereas in the former approach they are not.

The starting point for this survey is the observation that there is no consensus in engineering about the meaning of the term function itself, and that despite the broadly accepted value of a common and uniform language for functional reasoning, there is currently a series of functional reasoning models being proposed and developed that are distinct rather than convergent. Our aim was to make this lack of consensus explicit, primarily since we estimate that the existing differences between current approaches towards functional reasoning will eventually hamper the development of engineering computer tools and knowledge bases that can deal with functional reasoning.

The two steps that naturally succeed our analysis are, first, expanding our survey to other approaches towards functional reasoning, and, second, making an attempt to overcome the identified differences, either by integrating the approaches or by embracing some at the expense of others. The first step defines our next project, and also the second is one that we cannot take here. We finish instead with remarking that both options to overcome differences may in the end be one and the same. There is some room to integrate the three analyzed approaches.

Functions may be taken as intended abstracted behaviors represented by mappings of input flows to output flows, thus arriving at a general form-independent notion of function. The use of the repository in the Functional Basis approach may be shifted to the early phase of the decomposition of product functions into sub-functions, say by constraining the set of sub-functions allowed by the functional basis (see figure 1) to a set of sub-functions for which there are actually design solutions available in the repository [20], thus arriving at a functional reasoning scheme that at least takes into account current engineering knowledge about design solutions to functions. Yet, even if this scheme towards integration is viable, it will amount to a new approach to functional reasoning which is embraced at the expense of the three approaches we have analyzed.

ACKNOWLEDGMENTS

Research by Dingmar van Eck and Pieter Vermaas is supported by the Netherlands Organization of Scientific Research (NWO).

7. REFERENCES

[1] Stone, R. B. and Wood K. L., 2000, "Development of a Functional Basis for Design," *Journal of Mechanical Design*, Vol. **122**, pp. 359-370.

[2] Umeda, Y., Ishii, M., Yoshioka, M., Shimomura, Y., and Tomiyama, T., 1996, "Supporting Conceptual Design based on the Function-Behavior-State-Modeler," Artificial *Intelligence for Engineering Design, Analysis and Manufacturing*, Vol. **10**, pp. 275-288.

[3] Umeda, Y. and Tomiyama, T., 1997, "Functional Reasoning in Design," *IEEE Expert: Intelligent Systems and Their Applications*, Vol. **12**, pp. 42-48.

[4] Umeda, Y., Kondoh, S., Shimomura, Y., and Tomiyama, T., 2005, "Development of Design Methodology for Upgradable Products based on Function-Behavior-State Modeling," *Artificial Intelligence for Engineering Design, Analysis, and Manufacturing*, Vol. **19**, pp. 161-182.

[5] Chakrabarti, A. and Bligh, T. P., 1994, "An Approach to Functional Synthesis of Solutions in Mechanical Conceptual Design. Part1: Introduction and Knowledge Representation," *Research in Engineering Design*, Vol. **6**, pp. 127-141.

[6] Chakrabarti, A. and Bligh, T. P., 2001, "A Scheme for Functional Reasoning in Conceptual Design," *Design Studies*, Vol. **22**, pp. 493-517.

[7] Pahl, G. and Wallace, K., 2002, "Using the Concept of Functions to Help Synthesize Solutions," In: *Engineering Design Synthesis*, A. Chakrabarti (Ed.), Springer, London, pp. 109-119.

[8] Hirtz, J., Stone, R. B., McAdams, D. A., Szykman, S., and Wood, K. L., 2002, "A Functional Basis for Engineering Design: Reconciling and Evolving Previous Efforts," *Research in Engineering Design*, Vol. **13**, pp. 65-82.

[9] Chittaro, L. and Kumar, A. N., 1998, "Reasoning about Function and its Applications to Engineering," *Artificial Intelligence in Engineering*, Vol. **12**, pp. 331-336.

[10] Deng, Y. M., 2002, "Function and Behavior Representation in Conceptual Mechanical Design," *Artificial Intelligence for Engineering Design, Analysis, and Manufacturing*, Vol. **16**, pp. 343-362.

[11] Far, B. H. and Elamy, A. H., 2005, "Functional Reasoning Theories: Problems and Perspectives," *Artificial Intelligence for Engineering Design, Analysis, and Manufacturing*, Vol. **19**, pp. 75-88.

[12] Pahl G. and Beitz W., 1996 *Engineering Design: A Systematic Approach*, Springer, Berlin.

[13] Szykman, S., Racz, J. W., and Sriram, R. D, 1999, "The Representation of Function in Computer-based Design," *Proceedings of the 1999 ASME Design Engineering Technical Conferences*, DETC99/DTM-8742.

[14] Bryant, C. R., McAdams, D. A., Stone, R. B., Kurtoglu, T., and Campbell, M. I., 2006, "A Validation Study of an Automated Concept Generator Design Tool," *Proceedings of the 2006 ASME International Design Engineering Technical Conferences and Computers and Information in Engineering Conference*, DETC2006-99489.

[15] http://function.basiceng.umr.edu/delabsite/repository.html

[16] Umeda, Y., Takeda, H., Tomiyama, T., and Yoshikawa, H., 1990, "Function, Behaviour, and Structure," In: *Applications of Artificial Intelligence in Engineering, Proceedings of the fifth International Conference, Volume 1: Design*, J. S. Gero (Ed.), Springer, Berlin, pp. 177-193.

[17] Yoshioka, M., Umeda, Y., Takeda, H., Shimomura, Y, Nomaguchi, Y., and Tomiyama, T., 2004, "Physical Concept Ontology for the Knowledge Intensive Engineering Framework", *Advanced Engineering Informatics*, Vol. **18**, pp. 95-113.

[18] Ishii, M. and Tomiyama, T., 1996, "A Synthetic Reasoning Method based on a Physical Phenomenon Knowledge Base," In: *AI System Support for Conceptual Design*, J. Sharpe (Ed.), Springer, London, pp. 109-123.

[19] Hubka, V. and Eder, W., 2001, "Functions Revisited," In: *Proceedings of the 13th International Conference on Engineering Design, Glasgow, August 21-23, 2001*, pp. 69-76.

[20] Vermaas, P.E., 2007, "The Functional Modelling Account of Stone and Wood: Some Critical Remarks," In: *Proceedings of the 15th International Conference on Engineering Design, Paris, August 28-31, 2007*. Design Society, forthcoming.

