Artificial Intelligence for Engineering Design, Analysis and Manufacturing (2013), *27*, 271–279.
© Cambridge University Press 2013 0890-0604/13 $25.00
doi:10.1017/S0890060413000279

RESPONSE PAPER

A case for multiple views of function in design based on a common definition

AMARESH CHAKRABARTI,[1] V. SRINIVASAN,[2] B.S.C. RANJAN,[1] AND UDO LINDEMANN[2]
[1]Centre for Product Design and Manufacturing, Indian Institute of Science, Bangalore, India
[2]Institute of Product Development, Technical University of Munich, Munich, Germany

(Received March 1, 2012; Accepted March 1, 2013)

Abstract

Functions are important in designing. However, several issues hinder progress with the understanding and usage of functions: lack of a clear and overarching definition of function, lack of overall justifications for the inevitability of the multiple views of function, and scarcity of systematic attempts to relate these views with one another. To help resolve these, the objectives of this research are to propose a common definition of function that underlies the multiple views in literature and to identify and validate the views of function that are logically justified to be present in designing. *Function* is defined as a change intended by designers between two scenarios: before and after the introduction of the design. A framework is proposed that comprises the above definition of function and an empirically validated model of designing, extended generate, evaluate, modify, and select of state-change, and an action, part, phenomenon, input, organ, and effect model of causality (Known as GEMS of SAPPhIRE), comprising the views of activity, outcome, requirement–solution–information, and system–environment. The framework is used to identify the logically possible views of function in the context of designing and is validated by comparing these with the views of function in the literature. Describing the different views of function using the proposed framework should enable comparisons and determine relationships among the various views, leading to better understanding and usage of functions in designing.

Keywords: Common Definition; Environment; Extend Function; Generate; Intended Change; GEMS of SAPPhIRE; Requirement; Scenarios; Solution; System

1. INTRODUCTION

The purpose of designing, among others, is to achieve functions. Designers formulate functions, and to achieve these, they develop designs at various levels of abstraction. Knowledge of functions is important for modeling, generation, modification, exploration, visualization, explanation, evaluation, diagnosis, and repair of designs (Chakrabarti & Blessing, 1996; Stone & Chakrabarti, 2005).

In spite of years of research, several issues exist with the term *function*. It lacks a common meaning and definition; current definitions range from specific to generic (Keuneke, 1991; Umeda et al., 1996; Chandrasekaran & Josephson, 2000). Researchers attribute different meanings to function and use it interchangeably with behavior, purpose, and operation (Ullman, 1992; Chittaro & Kumar, 1998). The issues create confusion in communication and archiving, and obstruct teaching and formalization (Vermaas, 2011). Chakrabarti and Blessing (1996) attribute the difficulties to the multiple interpretations of function: abstraction or indexing of intended behavior, relationship between a design and its environment, external or internal behavior of a design, and so on. Further, there are no clear relationships among the various definitions of function (Stone & Chakrabarti, 2005; Kitamura & Mizoguchi, 2010).

The three main issues that hinder progress in research into functions are the following:

1. There is no clear and *overarching understanding* of what function is and why these apparently disparate research attempts should be called a research area with common goals and outcomes.
2. While there are multiple views of function, all of which seem useful in various contexts, no overall justification exists as to why these views are not just pragmatic attempts at solving the problems at hand but *theoretically inevitable* in designing.

Reprint requests to: Amaresh Chakrabarti, Centre for Product Design and Manufacturing, Indian Institute of Science, Bangalore 560012, India. E-mail: ac123@cpdm.iisc.ernet.in

3. With the exception of Crilly (2010), few have attempted to systematically relate these views to one another, as a means of improving communication across views and supporting functional reasoning across design contexts.

To help resolve these issues, the objectives of this paper are the following:

1. to review existing definitions of function and *propose a common definition* that underlies these (see Sections 2 and 3), and
2. to use an empirically, validated model of designing with the proposed definition to *identify a set of views* of function that should be present in designing, and to use the existing literature to validate their existence (see Sections 4 and 5).

2. VIEWS OF FUNCTION IN LITERATURE

Definitions and examples of function from the literature are reviewed in this section.

Rodenacker (1971) defines function as a *transformation from inputs (I) to outputs (O)*. The I/O are types of material, energy, or signal; for example, the function of a coffee mill is to convert coffee beans, electrical energy, and electrical signals into coffee powder, heat energy, and electrical signals.

Miles (1972) expresses function as *to do something*, for example, provide light, pump water, and indicate time. Miles distinguishes between primary and secondary functions, for example, the primary function of a domestic pump is to pump water and its secondary function is to operate at low noise level.

Pahl and Beitz (1977) define function as the *intended I/O relationship of a system* whose purpose is to perform a task. The task is I/O conversion; function is an abstraction of the task. The overall function is divided into subfunctions, which are connected to each other via I/O, with a temporal order. For example, in a fuel gauge, the overall function, which is to measure and indicate liquid quantities, is divided into subfunctions: receive, change, correct, channel, and indicate signal (Pahl & Beitz, 2007).

Hubka and Eder (1988) define function as the *required or desired, internal and cross-boundary, capabilities of a future or an existing system* to enable the system to perform its intended goals. They describe function as the transformation of desired and secondary inputs into desired and secondary outputs, within the system. Functions are categorized based on degrees of complexity, abstraction, and purpose.

Gero (1990) considers functions to *fulfill the expectations of the purposes* of the resulting artifact. For example, when designing windows, some functions are to provide daylight, to control ventilation, and so on. Gero identifies two kinds of behaviors (intended and actual), which are derived from function and structure, respectively. However, the actual behavior is not translated into function.

Keuneke (1991) defines function as the *intended purpose* of a device and describes the device's goals at an appropriate level of abstraction of interest at the device level. Functions are specified by the goals of activities of devices and components. Keuneke identifies *ToMake*, *ToMaintain*, *ToPrevent*, and *ToControl* as the types of expected functions that help achieve a specific partial state, achieve and sustain a desired state, keep a system out of an undesired state, and regulate state changes, respectively.

Ullman (1992) defines function as a *logical flow of I/O, as energy, material, or information*, between objects, or as a change of state of an object caused by these flows. He represents the overall function, decomposable into subfunctions, in terms of I/O, for example, the function of a one-handed bar clamp as transforming the grip force of one hand to a controllable force for clamping objects together.

Function is defined as a *description of behavior abstracted by humans* through recognition of behavior in order to utilize it (Umeda et al., 1996). The authors argue that it is difficult to represent function independent of the behavior from which it is abstracted and represent function as a combination of "to do something" and a set of behaviors that exhibit this. For instance, the behavior of "string oscillation" exhibits the function of "produce sound." Functions are causal and task decomposed into subfunctions (e.g., "generate light" into "generate electricity" and "light a lamp with electricity"). Note that the authors' function–behavior–state modeler helps detect unintended phenomena, but these are not abstracted into unintended functions.

Simon (1996) proposes that *fulfillment of a goal or adaptation to a purpose* of an artifact involves a relationship among the goal, the character of the artifact, and the environment in which it performs. For example, the purpose of a clock, "to tell time," can be described in terms of the gears, pendulum, spring forces, gravitational acceleration, and an environment with appropriate gravitational acceleration.

Chakrabarti et al. (1997) use *I/O transformation* (e.g., transform acceleration into voltage) for describing sensing functions (e.g., measure acceleration) that are satisfied using "solution principles" (e.g., using inertia effect). Effects are activated using "conceptual structures" that provide the properties and conditions necessary (Chakrabarti, 2001, 2004). Chakrabarti (1998) proposes two views of function: *intended behavior*, within, and *purpose*, outside, the system boundary. Intended behavior can be a subset of the *expected behavior*; for example, an intended behavior of a door latch is to press a door handle to retract a wedge, achieved by the expected behavior of a chain of components that transform the motion of the handle into that of the wedge. In purpose view, the door-latch function is to lock a door against wind.

Chittaro and Kumar (1998) define function as the *roles played by components in a system*; functions convey what components are needed to do (e.g., a switch in an electrical lamp is a barrier/conduit for electrical current). They take function as purposes that are *intentionally assigned* by designers or users to a system and its operational conditions.

Chandrasekaran and Josephson (2000) make a distinction between the environment- and device-centric views of function. In the environment-centric view, function is an *intended effect that a device has on its external environment*; for example, a buzzer's function is to provide a means by which someone at one location may cause sound to be produced at another location. In the device-centric view, a function is an *abstraction of the internal causal behaviors of the device*; for example, a buzzer's function is to make sound come from a box when a switch in another box is closed.

Wood and Greer (2001) define function as *what a product must do*. Function is represented as an action of the product on its inputs to produce outputs. I/O are material, energy, or signal. Product functions are decomposed into subfunctions to represent more elementary tasks (e.g., convert torque, transmit electricity). The functional basis ontology (Hirtz et al., 2002) use flows (I/O) and functions (operations on I/O) together to represent functions.

Deng (2002) identifies purpose and action functions, which are at different levels of abstraction. Action is an *interaction between two objects* (e.g., components of a system, or the system and its environment); action function is an *abstraction of intended behavior*. Purpose function is an abstract and subjective description of a designer's intention or the purpose of a design. The overall function and some high-level subfunctions are taken as purpose functions; the lower level functions that implement these are action functions. For example, the purpose function of a clock is "to tell time," which can be embodied by the action function "to rotate the hour, minute, and second hands about a pivot at a specific angular velocity."

Kitamura et al. (2006) define function of a device as a *role played by its behavior* to achieve a specific goal under a context of use, based on a certain capability inherent to the device. Kitamura and Mizoguchi (2010) distinguish between *actual* and *capacity* functions. Actual function is a *role played by a (device-oriented) behavior in a teleological context*; it is a thing that a device performs (cannot have), exists outside, and is less dependent on, the device. Capacity function is a *thing (property) that a device has (or is ascribed to a device)*, exists inside, and is dependent on, the device. *Artifact* and *device* functions are distinguished as follows: the first is an actual function intended by a user when an artifact is used externally by users and depends on users' intentions (e.g., the function of a screwdriver is to perform screwing or hammering function). Device functions are those performed by components in a system that contribute to the system's overall function and depend on the system's functional hierarchy. For example, a heat exchanger's function is to transfer heat, in a power plant whose function is to convert heat into electrical energy. The authors also distinguish between *essential* and *accidental* external functions of a system. The former is intended by designers of the system (e.g., a screwdriver's screwing function); the latter is not intended by them (e.g., a screwdriver's hammering function). Accidental functions are taken as affordances due to capacity functions. The properties that help perform an essential external function also afford the system to perform unanticipated, external functions.

Goel et al. (2009) define function as a transition from an intended input to an intended output, with a reference to behaviors that satisfy the function. For example, the function of a gyroscope is to convert an input angular momentum of some magnitude and direction into a proportional output angular momentum in the same direction.

The following views of function are identified from the above literature review:

a. *Level of abstraction view:* In the literature, functions are addressed at various levels of abstraction. For example, function in Miles (1972), Gero (1990), Keuneke (1991), and so on, are defined as to do something or the purpose of an artifact. Function in Rodenacker (1971), Hubka and Eder (1988), and so on, are defined as transformation of inputs to outputs. We consider the latter as less abstract because the purpose of a transformation is the intent behind, and an interpretation of, the transformation. Chakrabarti (1998), Deng (2002), and so on, explicitly distinguish functions using levels of abstraction.
b. *Requirement–solution view:* Functions are described as to do something, purpose, or to transform from input to output. Overall function in Pahl and Beitz (1977), Ullman (1992), and so on, is divided into subfunctions. Functions are therefore requirements to be fulfilled or solutions for fulfilling these requirements. Intent underlies both requirement and solution views; a proposal is not a solution unless it fulfills its requirements. In addition, a function-as-solution becomes a function-as-requirement at another level: subfunctions to achieve overall function are themselves requirements to be fulfilled. Note also that all functions are requirements, but not vice versa; the function of a shaft (to transfer motion) is a requirement, but a constraint on its size, a requirement, is not a function.
c. *System–environment view:* The literature describes functions as pertaining to the system or its effects on the environment. For example, functions in Gero (1990), purpose functions in Chakrabarti (1998), and so on, describe effects of a system on its environment. In contrast, secondary functions in Miles (1972), intended behavior in Chakrabarti (1998), and so on, focus on the system. In functions, relationships between system and environment are also described; for example, Rodenacker (1971) describes these in terms of I/O that transit the system boundary, while Chandrasekaran and Josephson (2000) describe these using modes of deployment. System hierarchy with relationships among subsystems are also used; for example, Pahl and Beitz (2007) and Ullman (1992), divide the overall function into subfunctions, with temporal, causal, and so on, relationships. Kitamura and Mizoguchi (2010) use system hierarchy between artifact and device functions that pertain, respectively, to system and system components.

d. *Designer–user view:* Functions in the literature are intended by designers or users; for example, the functions of Rodenacker (1971), Gero (1990), essential functions in Kitamura and Mizoguchi (2010), and so on, are intended by designers, while accidental functions in Kitamura and Mizoguchi (2010) are intended by users but unintended by designers.

3. PROPOSAL FOR A COMMON DEFINITION OF FUNCTION

The above definitions and views of function, we argue, have the following attributes in common:

a. *Function is always about intent:* It is about what a device *should* do or what effect it should have on its environment. For example, Miles (1972), Keuneke (1991), and so on, define function as purpose or to do something. Function in Gero (1990), purpose function in Chakrabarti (1998), and so on, are effects on an artifact's environment.
b. *Function is always about change:* It is a *change* between two scenarios: a current or anticipated but undesired scenario and a desired scenario. For instance, Rodenacker (1971), Stone and Wood (2000), and so on, define function as a transformation of inputs to outputs, where inputs would remain unchanged in the current/anticipated, undesired scenario, while these should change to outputs in the desired scenario. The change may also be to a desired scenario where the value of a parameter remains unchanged, from a scenario (before the device is introduced) where the value of that parameter keeps changing (e.g., "maintain room temperature") where the function is to change the current scenario of a room with variable temperature, to that with constant temperature.

We therefore define function of a design as *intended change* between two scenarios, before and after the introduction of the design. We argue that for an intended change to be a function, it must be *intended by designers*, because all changes intended by users but not by designers are *affordances*. For instance, even though a user may hammer an object using a screwdriver, one never says the function of a screwdriver is to hammer objects. It is possible for changes originally intended by users to be subsequently included by designers as intended in order to improve current designs. In such cases, these changes will become functions in the subsequent designs. Changes unintended by both designers and users are considered *side effects*. For instance, even if a clock goes slow when its battery is down, one never says the function of a clock is to run slow when its battery is down. Our definition of function is descriptive rather than normative, but it excludes functions described as "unintended."

Using this definition, several examples of function from the literature are now analyzed. "Operate at low noise level" (Miles, 1972) is an intended change between two scenarios: while existing pumps operate at high noise level (current scenario), the new pump should operate at low noise level (desired scenario). "*ToPreven*t" (Keuneke, 1991) is another example: it is an intended change, as a result of introducing the design that excludes an event (the one to be prevented) from the desired scenario, but it was included in the current scenario. Gero's example (1990) of a window to "Provide daylight" represents an intended change between two scenarios separated by introduction of the window from the expected, undesired scenario of a room with no daylight to the desired one of a room with daylight. Gero (1990) identifies two kinds of behavior, intended and actual, but *not* two kinds of function, which indicates that his work also agrees with function as encompassing *only* what is intended. Umeda et al. (1996) provide similar evidence, where unintended phenomena are identified but not abstracted into unintended functions.

4. A MODEL OF DESIGNING AND PROPOSED VIEWS OF FUNCTION

The various views of function (see Section 2) are important in various design contexts. In order to relate these to one another, we need not only a definition of function but also a model of designing that provides the contexts within which a function can be situated. For this, we use an empirically validated model of designing called the "extended GEMS of SAPPhIRE" model (e-GoS; Ranjan, 2012; Ranjan et al., 2012), comprising the views of activity, outcome, requirement–solution–information, and system–environment.

Srinivasan and Chakrabarti (2010) proposed GoS by integrating the views of activity, outcome, and requirement–solution. The activity view comprises the following activities: generate, evaluate, modify, and select (GEMS). The requirement–solution view comprises requirements and solutions at various abstraction levels. The outcome view uses the state-change, action, part, phenomenon, input, organ, and effect (SAPPhIRE) model of causality, which is described as follows. *Phenomenon* is an interaction between an entity and its surroundings; for example, heat transfer from a body to its surroundings. *State-change* is a change in property of the entity due to the interaction; for example, change in thermal energy stored in the body. *Effect* is the principle underlying the interaction; for example, the rate of heat transfer, $Q = h \times A \times \Delta T$, where h, A, and ΔT are the convective heat transfer coefficient, the surface area of the body, and the temperature difference between the body and surroundings, respectively. *Action* is a high level abstraction or interpretation of the interaction; for example, cooling of the body. *Input* is a physical quantity, in the form of material, energy, or signal, responsible for the interaction; for example, the temperature difference between the body and surroundings. *Organ* is a set of properties and conditions of the entity and its surroundings that are also responsible for the interaction; for example, the convective heat transfer coefficient. *Part* is a set of compo-

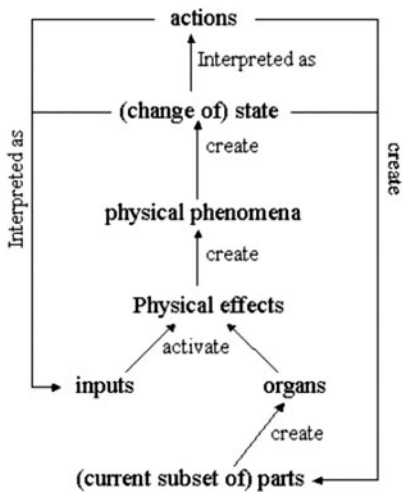

Fig. 1. The SAPPhIRE model of causality (Chakrabarti et al., 2005).

nents and interfaces comprising the entity and its surroundings; for example, a body held in air. *Parts* of an entity and its surroundings have *organs*. *Inputs*, together with the *organs*, activate *effects*, creating *phenomena*, which create *state-changes* that can be interpreted as *actions* (Fig. 1). The activity, outcome, and requirement–solution views are empirically validated for their abilities to describe activities, outcomes, requirements, and solutions in designing at multiple levels of abstraction (Srinivasan & Chakrabarti, 2010; Ranjan et al., 2013).

Ranjan et al. (2012, 2013) subsequently extended the GoS model by integrating to it the system–environment view, with the following constructs: *element*, *subsystem*, *system*, *environment*, and *relationships*. A *system* is the overall unit at any level of outcome abstraction, of which a *subsystem* is a subset. An *element* is a subset of a system or subsystem that cannot be further subdivided. An *environment* comprises all the subsets of the universe except the system. *Relationships* are how these are interlinked. Nidamarthi (1999) defines "related information as the information (facts or data), other than the contents of requirements and solutions, communicated through the design problem, colleague or any other source, concerning requirements or solutions," which is adapted in this requirement–solution–information view (Ranjan, 2012) as descriptions that are related to either requirements or solutions but are neither requirements nor solutions. Empirical validation shows that all these constructs are present in designing at various levels of abstraction of the SAPPhIRE model. According to e-GoS, during designing, GEMS activities are performed on SAPPhIRE outcomes, which evolve as requirements, solutions, or information of element, subsystem, system, environment, or relationship.

We propose the e-GoS and the common definition of function together as a *framework* for describing and integrating the various possible views of function. According to the framework, functions should be described using "requirements" (to represent intent) at any of the SAPPhIRE levels of abstraction, for an element, subsystem, system, environment, or relationship. Function at part level is not possible because it is a requirement that must be satisfied by a solution. Because part is the lowest abstraction level for a solution and because requirement must be at a higher abstraction level than that of its solution, the lowest possible abstraction level at which a function can exist is organ, one higher than the part level.

5. VALIDATION OF THE PROPOSED FRAMEWORK

The framework is validated by analyzing whether the views expressed in the literature (through examples) are included within the proposed framework. The validation is summarized in Table 1.

Rodenacker (1971) considers transformations as requirements intended by designers (coffee beans to coffee powder). His function pertains to the system and shows relationship with environment through I/O, which is expressible as "Input" of the SAPPhIRE model. The transformation of I/O (coffee beans to coffee powder) is expressible as state-change (nonpowder to powder) of the SAPPhIRE model. Rodenacker also describes transformation of electrical energy to heat (i.e., loss of energy as heat) as part of the function of a coffee mill. However, we conjecture that, because dissipation of energy could not have been intended as a function of this machine, this must have been accepted later because it could not have been avoided. Imagine a situation where users notice that absorption of heat dissipated by the coffee powder increases retention of its quality over a longer period of time: an affordance (Brown & Blessing, 2005). The designers of the coffee mill subsequently become aware of this and decide to include this as a function of the mill: to heat coffee powder. In that case, the functions of "convert electricity to heat" and "use heat to warm coffee powder" would both be regarded as changes intended by the designers, and hence functions according to the proposed framework.

All functions in Miles (1972) are requirements intended by designers, and hence functions. The primary (pump water) and secondary (operate at low noise level) functions in the examples pertain to the environment and system level, respectively; both are expressible using "Action" of the SAPPhIRE model.

Both overall function and subfunctions in Pahl and Beitz (2007) are requirements intended by designers. The overall function in the example (measure and indicate liquid quantities) belongs to the system level. The subfunctions (receive, change, correct, channel, and indicate signal) pertain to subsystem or element levels, depending, respectively, on whether they are further subdivided. The system and environment are linked using I/O crossing the system boundary; subfunctions are interrelated using temporal relationships, showing existence of relationships among functions. All the functions are expressible using "Action" of the SAPPhIRE model. The I/O of Pahl and Beitz are similar to "Input" of the SAPPhIRE model.

Table 1. *Description of functions using the identified views*

Literature	Level of Abstraction	Requirement–Solution	System–Environment	Designer–User
Rodenacker (1971)	State change, input	Requirement	System; system–environment relationship	IB-D
Miles (1972), primary function	Action	Requirement	Environment	IB-D
Miles (1972), Secondary function	Action	Requirement	System	IB-D
Pahl & Beitz (1977, 2007), overall function	Action	Requirement	System; system–environment relationship	IB-D
Pahl & Beitz (1977, 2007), subfunction	Action	Requirement	Sub system/element; relationships among and between elements and subsystems	IB-D
Hubka & Eder (2001, 2002), primary transformation	State change, input	Requirement	System; system–environment relationships	IB-D
Hubka & Eder (2001, 2002), secondary transformation	State change, input	Requirement	System; system–environment relationships	IB-D
Gero (1990), function	Action	Requirement	Environment	IB-D
Keuneke (1991), function	Action, state change	Requirement	environment, system, subsystem, element	IB-D
Ullman (1992), abstraction of transformation	Action, state change, input	Requirement	Environment, system, subsystem, element	IB-D
Goel et al. (2009), function	State change, input	Requirement	Environment, system, subsystem, element	IB-D
Umeda et al. (1996), function	Action	Requirement	Effect on environment	IB: D & U
Umeda et al. (1996), subfunction	Action	solution	Subsystem/element; relationships between and among subsystems and elements	IB: D & U
Simon (1996), function	Action	Requirement	System; system–environment relationship	IB-D
Chakrabarti et al. (1997)	Action, input, effect, organ	Requirement	System, subsystem, element, relationship	IB-D
Chakrabarti (1998), intended behaviour	Phenomenon	Requirement	System, subsystem, element	IB-D
Chakrabarti (1998), purpose function	Action	Requirement	Environment	IB-D
Chittaro & Kumar (1998)	State change, organ	Requirement	Subsystem, element	IB: D & U
Chandrasekaran & Josephson (2000), environment centric	Action	Requirement	Environment	IB-D
Chandrasekaran & Josephson (2000), device centric	Action, organ	Requirement	System	IB-D
Stone & Wood (2000), product function	Action	Requirement	System	IB-D
Stone & Wood (2000), subfunctions	Action, state change, input and phenomenon	Requirement	Subsystem, element	IB-D
Deng (2002), purpose function	Action	Requirement	System	IB-D
Deng (2002), action function	Phenomenon, organ	Requirement	Subsystem, element	IB-D
Kitamura & Mizoguchi (2010), actual function	Action	Requirement	Environment	IB-D
Kitamura & Mizoguchi (2010), essential functions	—	Requirement	System	IB-D
Kitamura & Mizoguchi (2010), accidental function	Action	Requirement	System	UB-D
Kitamura & Mizoguchi (2010), artefact function	—	Requirement	System	IB-U
Kitamura & Mizoguchi (2010), device function	—	Requirement	Subsystem, element	IB-D

Note: IB, intended by; UB, unintended by; D, designers; U, users.

Hubka and Eder (1988) consider functions as intended requirements and transformation of intended primary and secondary I/O. The primary and secondary functions are at the system level because functions are taken as transformation of I/O entering and leaving the system, respectively. The system–environment relationship is explained using I/O, all of which transit the system boundary. The I/O are expressible using "Input" and their transformation using "State-change" of the SAPPhIRE model.

Gero (1990) treats functions as requirements intended by designers. The examples (provide daylight and control ventilation) are at the environment level and are expressible using "Action" of the SAPPhIRE model.

Keuneke's functions (1991) are requirements intended by designers. The examples (*ToMake*, *ToMaintain*, *ToPrevent*, and *ToControl*) can pertain to any level within the system–environment view. Keuneke's functions and states are similar, respectively, to "Action" and "State-change" of the SAPPhIRE model.

Ullman (1992) defines (sub)function as transformation of I/O and the state-changes it enables; for example, the one-handed bar clamp function, transforming a single-handed grip-force into a controllable clamping-force, has I/O, transformation, and resulting state-change, all requirements intended by designers. I/O is similar to "Input" and the state-change (clamping objects together) is similar to "Action" of the SAPPhIRE model. The example transformation pertains to system; the state-change pertains to environment. Systems interact with environment through I/O. Ullman divides an overall transformation into subfunctions, suggesting the presence of system hierarchy for functions.

Function in Goel et al. (2009) is taken as a requirement intended by designers. Function (a transition between I/O) can be described using "State-change" in the SAPPhIRE model; the function in the example (gyroscope taking an input angular-momentum to deliver a proportional, output angular-momentum) is at the environment level. At the behavior level, the overall transition from I/O is divided into a series of state transitions. This shows breaking down of the overall function into subtasks, which can be seen as I/O transitions of subsystems and elements within the system.

Because functions in Umeda et al. (1996) are interpreted from behaviors by humans, functions are taken as requirements intended by designers or users. The overall function in the example (produce sound) pertains to the environment level and the subfunction (oscillate string) at the system level; these are expressible, respectively, using "Action" and "Phenomenon" of the SAPPhIRE model. Decomposition of required functions (generate light) into subfunctions (generate electricity and light lamp with electricity) shows the presence of functions at subsystem or element levels with various relationships among them.

Simon (1996) considers functions as requirements intended by designers. The overall function of a clock in his example, to tell time, can be described using "Action" of the SAPPhIRE model. Simon argues that artifacts are designed to perform only in particular environment(s), signifying the relationship between system- and environment-level functions.

Sensing functions in Chakrabarti et al. (1997) and Chakrabarti (2004) are at various abstraction levels in the SAPPhIRE model, Action (measure acceleration), Input (transform acceleration to voltage), Effect (transform acceleration to force using inertia effect), and Organ (the properties and conditions needed to satisfy the effects), and can be at various system–environment levels. Both views in Chakrabarti (1998) treat functions as requirements intended by designers. The purpose (lock door against wind) and the intended behavior (press a door handle to cause a wedge to retract) views are, respectively, at the environment and the system/subsystem/element levels and are expressible, respectively, using the "Action" and "Phenomenon" levels of the SAPPhIRE model.

Functions in Chittaro and Kumar (1998) pertain to components in a system, equivalent to functions of subsystems and elements in the proposed framework. Because functions are roles of components, they are requirements intended by designers or users. The overall function of an electrical switch in the example (turn on/off a lamp) is at the environment level and expressible using "State-change" of the SAPPhIRE model; the subfunctions (barrier or conduit for electrical-current) are "Organ"-level descriptions at the subsystem or element level.

The environment-centric function of Chandrasekaran and Josephson (2000) pertains to the environment level (buzzer's functions, to provide a means by which a person at one location may cause sound to be produced at another). Its device-centric function (to make sound come from a box when a switch in another box is closed) pertains to the system level. Both views are requirements intended by designers and expressible respectively, using the "Action" and the "Action and Organ" levels of the SAPPhIRE model.

Product-function in Stone and Wood (2000) pertains to the overall system and is a requirement intended by designers. This function (transmit torque) can be described using "Action" of the SAPPhIRE model. The subfunctions satisfying the overall product-function pertain to subsystems and elements. Subfunctions (e.g., in functional basis ontology) are expressible using "Action," "State-change," "Input," and "Phenomenon" of the SAPPhIRE model.

Purpose-functions (indicate time) in Deng (2002) involve human interpretations and thus are requirements intended by designers or users; they are expressible using "Action" of the SAPPhIRE model and pertain to the environment. Action functions (rotate, constant angular velocity, etc.) are requirements intended by designers, at subsystem and element levels, and are expressible using "Phenomenon" (rotate) and "Organ" (constant angular velocity) of the SAPPhIRE model.

Actual-functions (screwing function) in Kitamura and Mizoguchi (2010) are at the environment level (engaging a screw with the wall) and are expressible using "Action" of the SAPPhIRE model. Capacity functions resemble function at a lower level than actual-functions because these are used to satisfy actual functions. Artifact functions are at the system level; device functions are at the subsystem or element level. Both are affordances because they are intended by users. Essential functions are requirements of systems intended by designers. Accidental functions are unintended by designers. Functions unintended by designers can mean intended or unintended by users. Functions unintended by both designers and users are not functions but side effects.

6. DISCUSSION

Little has been proposed in the literature to resolve the confusion in understanding and using functional descriptions. An exception is Vermaas (2013), who proposes several possibilities to resolve the difficulties in describing functions. The common definition of function proposed in this paper is along the lines of the third possibility proposed by Vermaas: to propose an overarching concept of function for accommodating the coexisting descriptions of function. The proposed definition encompasses views of multiple, coexisting, and apparently disparate descriptions of functions from the literature.

Analysis of the literature in Section 5 using the proposed framework shows that all the functions reviewed in this paper can be described using the following: all abstraction levels of the SAPPhIRE model except parts, all levels of the system–environment view, and requirements from the requirement–solution–information view because the requirement view alone is sufficient for representing both the requirement and the solution views, as explained in Section 2. The activity view is not required for describing functions. Using empirical studies, Eckert (2013) identifies functions as being intentional and belonging to different levels of abstraction. These are similar to the requirement view and the level of abstraction view of functions, interpreted from the literature and reported in Section 2.

Analyses of the literature demonstrate that the various views on functions reported in the literature are not arbitrary but intended changes at every possible level of outcome and system abstraction that constitute designing; we argue that these enable designers to proceed from the highest (intended changes on the environment) to the lowest (intended organs to be provided by the system and its environment) level of abstraction. Together, these views, interlinked via the e-GoS model of designing, provide a coherent and interconnected tapestry of intents that a designer must formulate and satisfy in order to develop systems that have the desired effects on their environment.

Using a common, logically defendable platform to describe, compare, and determine relationships among the various descriptions of function has rarely been attempted in the past. Describing functions from the literature using a common framework, as attempted in this paper, is a step in that direction.

7. SUMMARY AND CONCLUSIONS

The literature reports many views of function, categorized using *level of abstraction*, *requirement–solution*, *system–environment*, and intention by *designer-user*. A common, overarching definition of function and a framework that combines this definition and an empirically validated model of designing are proposed; these are used to predict the set of views of function that are logically possible in designing. The framework is validated by comparing the views found in the literature with the predicted set of views. Validation reveals that all the views found in the literature are included in the predicted set. We conclude, therefore, that the framework provides a theoretical justification for coexistence of these views.

This demonstrates the potential of the framework as an enabler for comparison and integration of the various descriptions of function and paves way for better understanding and usage of functions in designing.

REFERENCES

Brown, D., & Blessing, L. (2005). The relationship between function and affordance. *Proc. Int. Design Engineering Technical Conf. and Computers and Information in Engineering Conf., DETC/CIE*, Report No. DECT2005-85017, Long Beach, CA, September 24–2008.

Chakrabarti, A. (1998). Supporting two views of function in mechanical design. Workshop on Functional Modeling and Teleological Reasoning. *Proc. 15th Association for the Advancement of Artificial Intelligence National Conf. AI*, Madison, WI, July 26–30.

Chakrabarti, A. (2001). Improving efficiency of procedures for compositional synthesis using bidirectional search. *Artificial Intelligence for Engineering Design, Analysis and Manufacturing 15(1)*, 67–80.

Chakrabarti, A. (2004). A new approach to structure sharing. *Journal of Computer and Information Science in Engineering 4(1)*, 11–19.

Chakrabarti, A., & Blessing, L. (1996). Representing functionality in design [Guest Editorial]. *Artificial Intelligence for Engineering Design, Analysis and Manufacturing 10(4)*, 251–253.

Chakrabarti, A., Johnson, A., & Kiriyama, T. (1997). An approach to automated synthesis of solution principles for micro-sensor designs. *Proc. Int. Conf. Engineering Design, ICED97*, pp. 125–128, Helsinki.

Chakrabarti, A., Sarkar, P., Leelavathamma, B., & Nataraju, B. (2005). A functional representation for aiding biomimetic and artificial inspiration of new ideas. *Artificial Intelligence for Engineering Design, Analysis and Manufacturing 19(2)*, 113–132.

Chandrasekaran, B., & Josephson, J. (2000). Function in device representation. *Engineering With Computers 16*, 162–177.

Chittaro, L., & Kumar, A. (1998). Reasoning about function and its applications to engineering. *AI in Engineering 12(4)*, 331–336.

Crilly, N. (2010). The roles that artefacts play: technical, social and aesthetic function. *Design Studies 31(4)*, 311–344.

Deng, Y. (2002). Function and behavior representation in conceptual mechanical design. *Artificial Intelligence for Engineering Design, Analysis and Manufacturing 16(5)*, 343–362.

Eckert, C. (2013). That which is not form: the practical challenges in using functional concepts in design. *Artificial Intelligence for Engineering Design, Analysis and Manufacturing 27(3)*, 217–232 [this issue].

Gero, J. (1990). Design prototypes: a knowledge representation schema for design. *AI Magazine 11(4)*, 26–36.

Goel, A., Rugaber, S., & Vattam, S. (2009). Structure, behavior, and function of complex systems: the structure, behavior, and function modeling language. *Artificial Intelligence for Engineering Design, Analysis and Manufacturing 23(1)*, 23–35.

Hirtz, J., Stone, R.B., McAdams, D.A., Szykman, S., & Wood, K.L. (2002). A functional basis for engineering design: reconciling and evolving previous efforts. *Research in Engineering Design 13(2)*, 65–82.

Hubka, V., & Eder, W. (1988). *Theory of Technical Systems: A Total Concept Theory for Engineering Design*. Berlin: Springer–Verlag.

Keuneke, A. (1991). Device representation—the significance of functional knowledge. *IEEE Expert 6(2)*, 22–25.

Kitamura, Y., Koji, Y., & Mizoguchi, R. (2006). An ontological model of device function: industrial deployment and lessons learned. *Journal of Applied Ontology 1(3–4)*, 237–262.

Kitamura, Y., & Mizoguchi, R. (2010). Characterizing functions based on ontological models from an engineering point of view. *Proc. Formal Ontology in Information Systems (FOIS)*, pp. 301–314. Amsterdam: IOS Press.

Miles, L. (1972). *Techniques of Value Analysis and Engineering*. New York: McGraw–Hill.

Nidamarthi, S. (1999). *Understanding and supporting requirement satisfaction in the design process*. PhD Thesis. University of Cambridge.

Pahl, G., & Beitz, W. (2007). *Engineering Design: A Systematic Approach*. London: Springer–Verlag.

Pahl, G., & Beitz, W. (1977). *Konstruktionslehre*. Berlin: Springer.

Ranjan, B.S.C. (2012). *An extended, integrated model of designing*. Masters Thesis. Indian Institute of Science, Bangalore.

Ranjan, B.S.C., Srinivasan, V., & Chakrabarti, A. (2012). The extended, integrated model of designing. *Proc. Tools and Methods of Competitive Engineering 2012*, Karlsruhe, Germany, May 7–11.

Ranjan, B.S.C., Srinivasan, V., & Chakrabarti, A. (2013). System–environment view in designing. *CIRP Design: 2012 Sustainable Product Development*, pp. 59–70. London: Springer–Verlag.

Rodenacker, W. (1971). *Methodisches konstruieren*. Berlin: Springer–Verlag.

Simon, H. (1996). *The Sciences of the Artificial*. Cambridge, MA: MIT Press.

Srinivasan, V., & Chakrabarti, A. (2010). An integrated model of designing. *Journal of Computer and Information Science in Engineering 10(3)*, 031013.

Stone, R., & Chakrabarti, A. (2005). Engineering applications of representations of function [Guest Editorial]. *Artificial Intelligence for Engineering Design, Analysis and Manufacturing 19(2)*, 63.

Stone, R., & Wood, K. (2000). Development of a functional basis for design. *Journal of Mechanical Design 122(4)*, 359–370.

Ullman, D. (1992). *The Mechanical Design Process*. Boston: McGraw–Hill.

Umeda, Y., Ishii, M., Yoshioka, M., Shimomura, Y., & Tomiyama, T. (1996). Supporting conceptual design based on the function–behavior–state modeler. *Artificial Intelligence for Engineering Design, Analysis and Manufacturing 10(4)*, 275–288.

Vermaas, P. (2011). Accepting ambiguity of engineering functional descriptions. *Proc. Int. Conf. Engineering Design, ICED11*, pp. 98–107. Copenhagen: Design Society.

Vermaas, P.E. (2013). The coexistence of engineering meanings of function: four responses and their methodological implications. *Artificial Intelligence for Engineering Design, Analysis and Manufacturing 27(3)*, 191–202 [this issue].

Wood, K., & Greer, J. (2001). Function-based synthesis methods in engineering design. In *Formal Engineering Design Synthesis* (Antonsson, E., & Cagan, J., Eds.). New York: Cambridge University Press.

Amaresh Chakrabarti is a Professor at the Centre for Product Design and Manufacturing, Indian Institute of Science; Coordinator for the Innovation, Design Study and Sustainability Laboratory; and cofounder of the Virtual Reality Laboratory at the Centre for Product Design and Manufacturing. He received his PhD in engineering design from the University of Cambridge in 1992, his masters in mechanical systems design from the Indian Institute of Science in 1987, and his bachelors in mechanical engineering from the University of Calcutta in 1985. He led the design synthesis team for 10 years at the Engineering Design Centre, an Engineering and Physical Sciences Research Council Centre for Excellence in Engineering Design in the University of Cambridge's Department of Engineering. Prof. Chakrabarti's research interests include design synthesis, creativity, ecodesign, knowledge capture and reuse, collaboration, requirements engineering, computer-aided design, and design for variety.

V. Srinivasan has been a Postdoctoral Researcher at the Institute of Product Development, Technische Universität München, since mid-2011. He received his PhD in engineering design from the Indian Institute of Science in 2011 and his bachelors in mechanical engineering from the University of Madras in 2003. Dr. Srinivasan's research interests include design creativity, design theory and methodology, functional modeling, design synthesis, biomimetics, conceptual design, and engineering design.

B.S.C. Ranjan is a PhD student in the Innovation, Design Study and Sustainability Laboratory, Centre for Product Design and Manufacturing, Indian Institute of Science. He received his masters in engineering design from the Indian Institute of Science in 2012 and his bachelors in mechanical engineering from Visvesvaraya Technological University in 2007. His research interests include design methodologies, product design, embodiment design, machine tools, robotics, automotive engineering, and biomimetics.

Udo Lindemann has been a Professor and Head of the Institute of Product Development, Technische Universität München since 1995. He received Dr.-Ing from Technische Universität München in 1979 and Dipl.-Ing from Technische Universität Hannover in 1974. He has 15 years of diverse experience in industries like RENK AG and MAN Miller Printing Machines GmbH. Prof. Lindemann's research interests include design theory, methods, and product design in the following: innovative and creative design, design representations, creative problem solving, product design methodologies, integrated product development, industrial design, entrepreneurship, design to cost, knowledge management, cooperation and collaboration in distributed teams, lean development, structural complexity, change management, and strategic planning.