

An Ontological Model of Device Function and Its Deployment with Extensions for Engineering Knowledge Sharing

Yoshinobu Kitamura, Yusuke Koji and Riichiro Mizoguchi

*The Institute of Scientific and Industrial Research, Osaka University,
8-1 Mihogaoka, Ibaraki, Osaka, 567-0047 Japan
{kita, koji, miz}@ei.sanken.osaka-u.ac.jp*

Abstract. Functionality is one of the key concepts of knowledge about artifacts. Functional knowledge shows a part of designer’s intension (so-called design rationale), and thus its sharing among engineers plays a crucial role in team-activities in engineering practice. Aiming at promoting engineering knowledge management focusing on functional knowledge, we have developed an ontological modeling framework of functional knowledge, which includes an ontology of device and function as conceptual constraints for capturing functions and a functional concept ontology as a controlled vocabulary. This framework has been successfully deployed in a manufacturing company in daily engineering activities. This paper reports on theoretical deepening of the ontologies and two practical extensions based on the experience of the deployment. The first theoretical part of this paper discusses some ontological issues of functionality of artifacts and redefines function as a role. The second part of this paper discusses lessons learned in the deployment and two extensions based on them. One extension is a lexical layer for functional terms in order to help engineers select appropriate functional concepts and to facilitate use of domain-specific terms familiar to them. The other extension is establishment of ontological modeling guidelines, which help engineers commit to the ontologies and describe models compliant with the ontologies..

Keywords: Engineering domain, Functionality, Knowledge management, Engineering Design, Role

1. Introduction

Functionality is one of the key concepts of domain knowledge about artifacts. While there is no common understanding of definition of function¹ and its representation, it is widely accepted that function of an artifact is tightly related to intention of designers or users (de Kleer, 1984; Chandrasekaran *et al.*, 1993; Umeda & Tomiyama, 1997; Chittaro & Kumar, 1997; Hubka & Eder, 2001, Stone & Chakrabarti 2005). Intuitively, a function of a product explains what users can get using it, i.e., effects or utility of the artifact. A function of a component embedded in a system explains how it contributes to achieving the system’s whole-function (so-called “how things work”). Recognition of a function of an artifact is dependent on a system, an environment, a situation, or a usage, in contrast to objective data such as shape and structure.

In engineering situations, functional knowledge is an important constituent for engineering knowledge management. It plays a crucial role in many engineering activities such as designing, manufacturing and maintenance. Especially, conceptual design can be viewed as an activity to develop a functional structure (i.e., relationships among functions of components and a system) and mapping relations between functions and physical structures which can realize them. Functional knowledge represents a part of designer’s intention so-called *design rationale* (DR) (Chandrasekaran *et al.*,

1993; Lee 1997), which gives justification of the current design including reasons of existence of a component or a sub-system in the system. Explicit representation of DR facilitates engineering activities such as design review, product improvement, and facility maintenance. For example, in design review to doublecheck an original design by a team of designers, an explicit description of the original designer’s intentions helps other people understand the original design more effectively. In facility maintenance, in order to adjust a working parameter of the facility, a maintenance engineer should understand the reason of the current value which is a result of the designer’s decisions.

Nevertheless, it is well-known that sharing of such knowledge about function is difficult in practice, which has been confirmed by our experience of collaborative research with a production company. Few CAD/CAM/PDM systems deal with such subjective knowledge. Thus, engineers have to rely on documents in natural language. Engineers have been regularly writing various kinds of technical reports/documents and have stored much of those in databases. Unfortunately, however, few such technical documents have been efficiently reused. One of the reasons of this difficulty is lack of *semantic constraints* for functional knowledge. By the semantic constraints we mean here that guideline or restriction which helps a knowledge-author describe models compliant with a specification of conceptualization to which the author commits. Without such semantic constraints, functional models tend to be ad hoc, specific to the target product, and hence not reusable. Although much research has been conducted on the

¹ In this article, the term ‘function’ is used as the same as ‘functionality’.

representation of functionality in functional representation (Sembugamoorthy & Chandrasekaran, 1986; Keuneke, 1991, Chittaro *et al.*, 1993; Lind 1994, Umeda *et al.*, 1996), engineering design (Pahl & Beitz 1988; Gero, 1990; Hubka & Eder, 1988, 2001, Hirtz *et al.*, 2002) and value engineering (Miles 1961), there is no common definition of the concept of function itself (Umeda & Tomiyama, 1997; Chittaro & Kumar, 1997; Hubka & Eder 2001) and clear categorization of relationships among functions. Such consideration is not enough for deriving effective guidelines as semantic constraints. We argue that ontologies of functionality can provide semantic constraints on knowledge-contents as ‘meta knowledge’. We will discuss its needs in the next section.

The goal of our research is to facilitate engineering knowledge management of functional knowledge by providing an ontological modeling framework as such a semantic constraint. Ontology about functionality aims at specifying a *conceptual viewpoint* for capturing the target world and a *controlled vocabulary* to describe the knowledge at an appropriate level of abstraction. Much work on engineering domain ontologies has been done (Cutkosky, *et al.*, 1993; Gruber & Olsen 1994; Borst *et al.*, 1997; López, *et al.*, 1999; Yoshioka *et al.*, 2004). Many of them aim at improvement of interoperability for communication among agents or tools (Cutkosky, *et al.*, 1993; Yoshioka *et al.*, 2004). Rather, we aim at “ontology as meta knowledge” as discussed above. One remarkable example of meta-knowledge type is the PhysSys ontology (Borst *et al.*, 1997). It, however, has no ontology for functions from the teleological viewpoint.

The authors have been involved in ontology-based modeling of physical systems for many years and have established an ontological framework for functional knowledge (Kitamura *et al.*, 2002; Kitamura & Mizoguchi, 2003; 2004a). This framework includes an ontology of device and function as conceptual viewpoint and a functional concept ontology as a controlled vocabulary. A knowledge management software called SOFAST based on the framework has been successfully deployed in a manufacturing company in Japan for sharing functional knowledge (Kitamura *et al.*, 2004).

This paper reports on extensions of these previous efforts. This paper’s contribution is twofold. The first part of this paper discusses ontological issues of functionality of artifacts. It investigates fundamental issues as requirements of an ontology about function based on the concept of “role” in Ontological Engineering. Then, we refine our previous definition of function reported in (Kitamura *et al.*, 2002).

The second part of this paper discusses lessons learned in the deployment and two extensions based on them. In the deployment, it turned out that engineers have

difficulties in selecting appropriate generic types of functions and describing functional models compliant with the ontologies. The extensions to reduce these difficulties are a lexical layer for functional terms, and ontological modeling guidelines, respectively. The former provides multiple mapping relations between types of functions and superficial labels (names) for denoting the types. A new version of SOFAST had been developed in order to help engineers select a function when they describe functional knowledge, and to search a function independently of difference of superficial labels. The latter provides knowledge authors with a checklist for revising their functional models in order to describe models compliant with the ontologies.

This paper is organized as follows. Section 2 discusses needs of ontologies for functionality. Section 3 discusses fundamental issues as requirements of an ontology about function. Section 4 presents our ontologies about functions. The experience in the deployment is discussed in Section 5. Section 5.1 summarizes the process of the deployment and the usages discussed in (Kitamura *et al.*, 2004). The effects of the ontologies are discussed in Section 5.2. Section 5.3 discusses the lessons learned in the deployment, which includes success factors of the deployment and difficulties faced in the deployment. Section 6 presents the extensions to reduce the difficulties. Then, related work is discussed followed by some concluding remarks.

2. Current Challenges in Functional Knowledge Modelling and Ontology as its Solution

This section presents two examples that demonstrate the difficulty in functional knowledge modeling and then proposes use of ontologies as a solution. Firstly, functionality in Value Engineering is represented in “verb+noun” style (Miles, 1961) and on the basis of this, one might describe “to weld metals” as a function of a welding machine. However, “to weld metals” implies both the metals are joined and their parts are fused. From the viewpoint of functionality in manufacturing, joining is only the goal the designer intends to attain (“what to achieve”), while the fusion can be regarded as a characteristic of “how to achieve that goal”. In fact, the same goal, say, “to join”, can be achieved in different ways (e.g., using nuts & bolts) without the fusion. If a function of the welding machine is described as “to join”, the commonality between two facilities can be found. This issue is not a terminological but *ontological* in order to distinguish “what to achieve” from “how to achieve”. We distinguish terminological problems, for instance, to use a word “to fix” instead of “to join” for representing the same concept. Thus, this example demonstrates the importance of functionality in reusable functional knowledge.

The well-known systematic design methodology in (Pahl & Beitz 1988), on the other hand, includes hierarchical structures of functionality based on input-output relations (so-called functional decomposition). However, it is not easy to describe such functional models. For the welding machine example, one might describe “to put objects together”, “to make an arc”, and “to leave them” as sub-functions (decomposed micro-functions) of the goal function “to join”. These sub-functions certainly describe decomposition of the input-output relation. However, there is an implicit intermediate function “to heat objects” between “to make an arc” and the goal function. In fact, “to heat objects” can be achieved by “to make current flow” instead of “to make an arc”. Moreover there is another implicit function “to melt objects” as the goal function of the heating function. This second example demonstrates the importance of practical specifications on semantics for functional decomposition, in addition to standard specifications as decomposition of input-output.

These suggest the necessity for semantic specifications for the content of functional knowledge, which help knowledge authors describe knowledge compliant with a conceptualization. It can be specified as ontologies (Mizoguchi, 2005), i.e., “explicit specifications of conceptualization” (Gruber 1993). Ontology about functionality specifies a *conceptual viewpoint* and a *controlled vocabulary* for functional knowledge. The former provides guidelines or constraints on modeling, which helps knowledge authors to describe functional knowledge consistently, and especially to distinguish “what to achieve” from “how to achieve”. On the other hand, the *controlled vocabulary* provides a systematized set of generic verbs representing functionality of devices at an appropriate level of abstraction.

3. Ontological Issues of Function

This section discusses ontological issues as requirements for reusable and consistent functional knowledge. Our goal here is to define function, generic functions, and relationships between functions clearly and operationally.

3.1 Definition of Function with Behavior

For clear definitions of functionality, the relationship with “behavior” plays a crucial role. The distinction between function and behavior originates from the qualitative reasoning (QR) research such as (de Kleer & Brown 1984). *Behavior* of a device here represents temporal changes of physical quantities of a physical entity (we call *operand*) other than the device. It is objective and independent of the *context* which includes designer’s intention, user’s aims and the system in which the entity is embedded. In QR, in order to realize reusability and composability of the component model, context-dependent information is

carefully excluded (called No-Function-In-Structure principle²) (de Kleer & Brown 1984).

In comparison with behavior, *function* is related to the intention of a designer or a user (i.e., teleological) and hence context-dependent. A behavior can perform different functions according to the context. For example, a heat exchanger can be used as a heater or a radiator. The behavior is the same in any context, that is, a heat flow from the warmer fluid to the colder one, where the flows of fluid are operands. The functions of the heater and the radiator can be “to give heat” and “to remove heat”, respectively. This difference of functions is dependent on the embedded system.

Thus, the first issue is to clarify this *teleological interpretation relation* between behavior and function and to define functions based on this relationship. In the literature, this relationship is defined as “means and ends” (Lind 1994), F-B relationship (Umeda *et al.*, 1996), “aims-means” (Hubka & Eder 1988) (This includes design requirements as well) or causal patterns (de Kleer 1984). On the other hand, the well-established standard taxonomy, called *functional basis*, for functions by the NIST Design Repository Project (Hirtz *et al.*, 2002) lacks clear relationship with objective behaviors based on such teleological relationship.

In order to clarify the relationship and to define functions operationally, our approach is to describe a function as “behavior plus information for teleological interpretation” in terms of a set of primitives (called *functional toppings*) as discussed in Section 4.1. Moreover, we define generic functions as constraints on the information as discussed in Section 4.2.

3.2 Function as a Role

The second issue related to the definition of function is the “*role*” concept in ontological engineering research. Intuitively, a *role* is something that can be *played by* an entity in a context. Precisely, in (Sowa, 1995), a role is the *secondness* concept which is dependent on a pattern of relationship. In (Masolo *et al.*, 2004), a role is *anti-rigid* (i.e., contingent (non-essential) property for identity), *dynamic* (temporary and multiple), and *founded* (i.e., extrinsic property defined with external concept). Similar to these definitions, by role we mean here such a concept that an entity plays in a specific *context* and cannot be defined without mentioning external concepts (Mizoguchi *et al.*, 2000; Kozaki *et al.*, 2002; Sunagawa *et al.*, 2006). We distinguish *role* (something to be played) from *role-holder* (something playing a specific role). This role-holder is different from *role-player* which can play a role. For example,

² Although a behavioral model depends on modeler’s assumptions and viewpoint for capturing the target world as the same as all information models do, the concept of “behavior” itself represents context-independent changes.

the “husband role” (a role concept) is played by a member x of the class “man” (the class to which the player of this role is constrained) in a “marriage” relation (this relation gives the role context). The entity x that plays the “husband role” is the “role holder” and is a member of “husband” (namely, the class-equivalent³ of man with husband role). Our ontology editor of an environment for building/using ontology named Hozo has a capability to deal with roles (Kozaki *et al.*, 2002; Sunagawa *et al.*, 2006).

Let us analyze the property of function. Firstly, a function (and a behavior as its basis) is *founded*, since a function of an artifact affects an entity (the operand) other than the artifact itself and causes temporal changes of such an entity (this is the *behavior of the device* in the sense of Section 3.1). For example, a radiator decreases temperature of the warmer fluid and the definition of the removing-heat function refers to the change of the warmer fluid’s temperatures as input and output. Thus, definition of a function of an artifact requires an operand as an external entity.

As discussed in Section 3.1, a *behavior* can perform different *functions* according to teleological contexts. By definition, a behavior is objective and thus does not lose identity by the change of function. Thus, a function is *anti-rigid*. Moreover, a function can be performed (realized) by different behaviors. A function is associated with specific constraints on a part of behaviors to realize the function. A behavior can perform multiple functions simultaneously. Thus, a function is dynamic and multiple.

A function can be defined as a *role played by behavior in a teleological context*. We say that “a behavior can play a function role”. If a device performs a behavior and the behavior plays a *function role* in a context, then the device plays a *function-performer role* in the context. For example, the heat-exchange behavior plays the removing-heat *function role* and then a heat exchanger plays the *function-performer role* of removing-heat as a radiator.

The teleological context for an engineering system can be determined according to designer’s intention or user’s intention (i.e., how to use it). The context for a component embedded in a system is configuration of the system (precisely, as discussed in Section 3.4 and 4.1, a functional structure).

In the literature, similar concepts are discussed. Chandrasekaran and Josephson (2000) use the concept of *role as natural* (without human’s intention) effects on environment (e.g., the role of cloud is to give rain) and define *function* as “role + intention”. In EPISTLE Framework, the concept of *facility* is defined as a functional thing, capability to perform a function and a service (West 2004). Fan *et al.* define *purpose* of an

artifact as *default role* which is expected to be done by the artifact (Fan *et al.*, 2001). Breuker and Hoekstra (2004) point out function as role and discussed *mental roles* in law. Garbacz (2005) points out that artifact function is *state of affairs*, which represents connection between objects and processes (in our terminology, operands and behavior). He identifies some upper-level functions based on DOLCE. Vermaas (2005) captures a function with relationship among an agent, a use plan, physicochemical capacity and a justifying account.

In summary, the second issue is to define the function as a role of behavior. Such definition requires a description of the context for functional interpretation. The *functional topping* is a localized representation of the context. Our definition of function as role of behavior with functional context is discussed in Section 4.1.

3.3 Device Ontology and Entity’s Roles

For consistent representation of functions, consistent representation of behavior and system is a very important issue as well. In the design literature, German systematic design approach (Pahl & Beitz 1988) has provided us with a basic viewpoint to capture functions as mentioned in Section 2, in which functions are regarded as the input-output relations of a black-box. The black-boxes can be connected and aggregated. Such a device-centered viewpoint originating from systems dynamics theory is called *device ontology* here. Similar ontologies have been established in the literature (de Kleer and Brown 1984; Gruber & Olsen 1994; Borst *et al.*, 1997). A device ontology is suitable as a basis for establishing ontologies of functions, since functions are usually considered as what components or devices achieve.

The definition of the device ontology also requires representation of “role”. Let us consider a manufacturing machine called a wire-saw, which is designed to slice a semiconductor ingot into wafers using a wire moved by rollers. To assign a role to this wire is not trivial. It exerts force on the ingots as a *device*, and is moved by rollers as *operand*. In order to assign roles to entities in a consistent manner, we have to maintain the relationship among entity, role and behavior. We will revisit this example in Section 5.2.

Thus, the third issue is to establish a *role assignment system* which helps model-authors assign roles to entities consistently. As discussed in Section 4.1, we extend the de Kleer’s device ontology (de Kleer and Brown 1984) by redefining *behavior*, *conduit* (a subtype of *device*) and *medium* (a subtype of *operand*) for a richer role assignment system.

3.4 “is-a” and “part-of” Relations of Function

The German systematic design approach (Pahl & Beitz 1988) provides a fundamental relationship

³ It is an abstraction of individuals (men) playing a specific role. See details (Kozaki *et al.*, 2002; Sunagawa *et al.*, 2006)

among function as well, so-called *function decomposition* as mentioned in Section 2. It represents how a function is achieved by a series of sub-functions which are finer-grained functions (we call “*is-achieved-by*” relation), which is a kind of so-called “part-of” relation. Usually (but not always), it corresponds to the whole-part (*aggregation*) hierarchy of physical structures of devices, that is, the whole system, sub-systems and components. In the literature, this relation has been captured as whole-part relation (Lind 1994), and “degree of complexity” (Hubka & Eder 1988) as well as the function decomposition (Pahl & Beitz 1988).

On the other hand, we can consider “*is-a*” relation among functions. It represents generic or abstract concepts of functionality. Pahl and Beitz (1988) defined highly-abstracted functions (called generally-valid functions). Hubka and Eder (1988) identified the hierarchy for the “degree of abstraction” of functions which represents the specialization of functions with additional conditions. The conditions, however, may sometimes (not always) include the characteristics of a specific method of achieving a function such as “transportation by sea” (Hubka & Eder 1988) which has the same difficulty as “welding” as discussed in Section 2.

It is not easy to distinguish the relationships among functions, that is, *part-of*, *is-a*, and the *teleological interpretation* (discussed in Section 3.1). One of the reasons is that whenever a function is achieved, its super-classes functions in the *is-a* hierarchy are also achieved. The papers such as (Sembugamoorthy & Chandrasekaran 1986) do not distinguish *teleological interpretation* from the *part-of* relation and thus define that “behavior” is how to achieve a function where distinction between behavior and function is relative.

In summary, the fourth issue is to clarify the relationships between functions such as *part-of* and *is-a* and to find descriptors of the *teleological interpretation*. We introduce the concept of *way of function achievement* as conceptualization of the *part-of* relation. Moreover, we define *is-a* relations among them as discussed in the next Section.

4. Ontologies for functional knowledge

This section explains an ontology of device and function (Section 4.1), a functional concept ontology (Section 4.2), and functional knowledge based on these ontologies (Sections 4.3 and 4.4).

4.1 An Ontology of Device and Function

Our ontologies have been defined in an ontology editor of an environment for building/using ontology named Hozo (Kozaki *et al.*, 2002, Sunagawa *et al.*, 2006). Figure 1 shows an example and Figure 2 shows a portion of an ontology of device and function. The ontology editor basically supports frame-based

representation with slots. Concepts are represented as frames (denoted by nodes in Fig. 1 and 2) with slots (right-angled link) and the *is-a* relations among concepts (straight link with “is-a”). Concepts are categorized into the wholeness concepts composed of part concepts and the relational concepts between the concepts. A wholeness concept has slots of part concepts (part-of relation denoted by right-angled link with “p/o”) and slots of attributes (“a/o”). A relational concept has slots of participant concepts (participate-in relation, denoted by “p/i”) and the attribute-slots. Figure 1 shows an example of definition of the husband role discussed in Section 3.2 (i.e., a *husband* (role holder) is defined as a *man* (class constraint) playing a *husband role*) in the “marital relation” concept. The upper right part of Fig. 1 shows its definition from a role-centered view. It is defined also in the “married couple” which is a wholeness concept corresponding to the “marital relation”.

The ontology of device and function consists of an extended device ontology and an ontology of function. Figure 2 includes a portion of an upper-level ontology, by which we intend not to claim a generic upper-level structure but to clarify the meaning of the concepts in the ontology of device and function.

As mentioned in the previous section, one of the key concepts is *behavior* which represents objective temporal changes of physical quantities of a physical entity. In Fig. 2(a), device’s *behavior* is defined as temporal change of another *physical-entity* called an *operand*. The *device* is defined as a role-holder played by a *physical-entity*, which operates on the *operand* and changes its *physical-attributes*. The *operand* is something that flows through the device and is affected by the device. Thus, it is defined as another role-holder in the *behavior*. The *operand role* can be played by fluid, energy, motion, force, or information. The temporal change of *operands* is represented as a pair of *physical-states* as input and output of the device, each of which represents a value of a *physical-attribute* at a *port* of the device. A *port* of a device is a virtual interface for propagation of physical-attributes’ values to another device. Each device is connected to each other through its input and output *ports* (Fig. 2(b)). Thus, the *behavior* represents objective conceptualization of its input-output relation

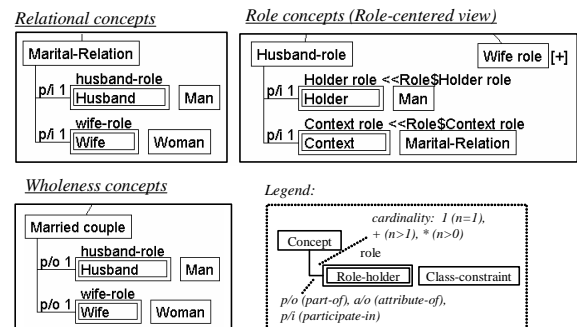


Fig. 1. Example of definition of a role concept (husband role) in Hozo

as a black box. From the viewpoint of causality, as shown in Figure 2(a), the changes in a behavior are caused by sub-behaviors (we call motion), which are a sequence of finer-grained behaviors.

The *behavior* is defined as a sub-type of *occurrent* (i.e., an entity has time-dependent part), *active* (i.e., change of attribute's value in certain time-interval) and *action* (i.e., process with an agent (i.e., actor, subject) who makes the process happen). Thus, it is distinguished from *continuant*, *stative* (i.e., a set of attribute-value pairs of an entity or entities at a time-point), and *phenomena* (i.e., process with participants without specific agent), respectively. Although it has rather general label “behavior”, it is the specialized for device-oriented modeling as black box with flowing entities.

We defined rich types of behavior, conduit, and medium. We categorized the meanings of behavior into four types (from B0 to B3) (Kitamura & Mizoguchi 2004a). The definition of behavior above (i.e., behavior based on *states* of the flowing *operands* at *ports* of a *device*) is called *B1-behaviour* in terminology in (Kitamura & Mizoguchi 2004a). A *conduit* (e.g., a pipe and a shaft) is defined as a special device that transmits an operand without any change in an ideal situation. A *medium* (e.g., steam for heat energy) is something that holds an operand and enables it to flow between devices. A medium role and a conduit role can be played by an entity simultaneously, although a device role and an operand role cannot be. Such multiple role-playing is a solution for the issue in Section 3.3. We revisit this issue in Section 5.2.

A (*base*-)function role is defined as a role concept which is played by a *behavior* under a *function-context*

(Fig. 2(c)). In the function-context, there is a *function-performing relation* among two physical entities and a behavior (Fig. 2(d)). In the relation, the behavior plays a base-function role, which is called a base-function role holder (Fig. 2(e)).

The *function-context* represents teleological situation for specific goals in which a function is defined. It has two subclasses; that is, *system-function-context* and *user-function-context* as shown in Fig 2(f). The system-function-context is for a function of a component embedded in a system. It can be determined by the system's function and functions of other components connected to the component. A component's function with the other component's functions (we call *method function*) achieve the system's function (*goal function*). In Fig. 2(g), the achievement relation is defined as a relational concept named “*way of function achievement*” relation. It has two *participant* slots; that is, a function with the *goal function role* and functions with the *method function role*. It is used in the definition of the *system-function-context* as a restriction, which defines that a base-function in the context should play a *method function role* for a *goal function*. (the third slot of the *system-function-context* shown in Fig. 2(f)). The base-function (a role holder) is inherited from the third slot of the *function-context* shown in Fig. 2(d), which defines that a behavior plays both *participating role* in the *function-context* and the *base-function role* in the *function-performing relation* discussed above.

These definitions are a more detailed version of our definitions in previous papers (Sasajima *et al.*, 1995; Kitamura *et al.*, 2002), that is, a function as a teleological interpretation of a *B1-behavior* under a *goal*. The “goal” in this previous version is redefined as the function-context in detail. The “interpretation”

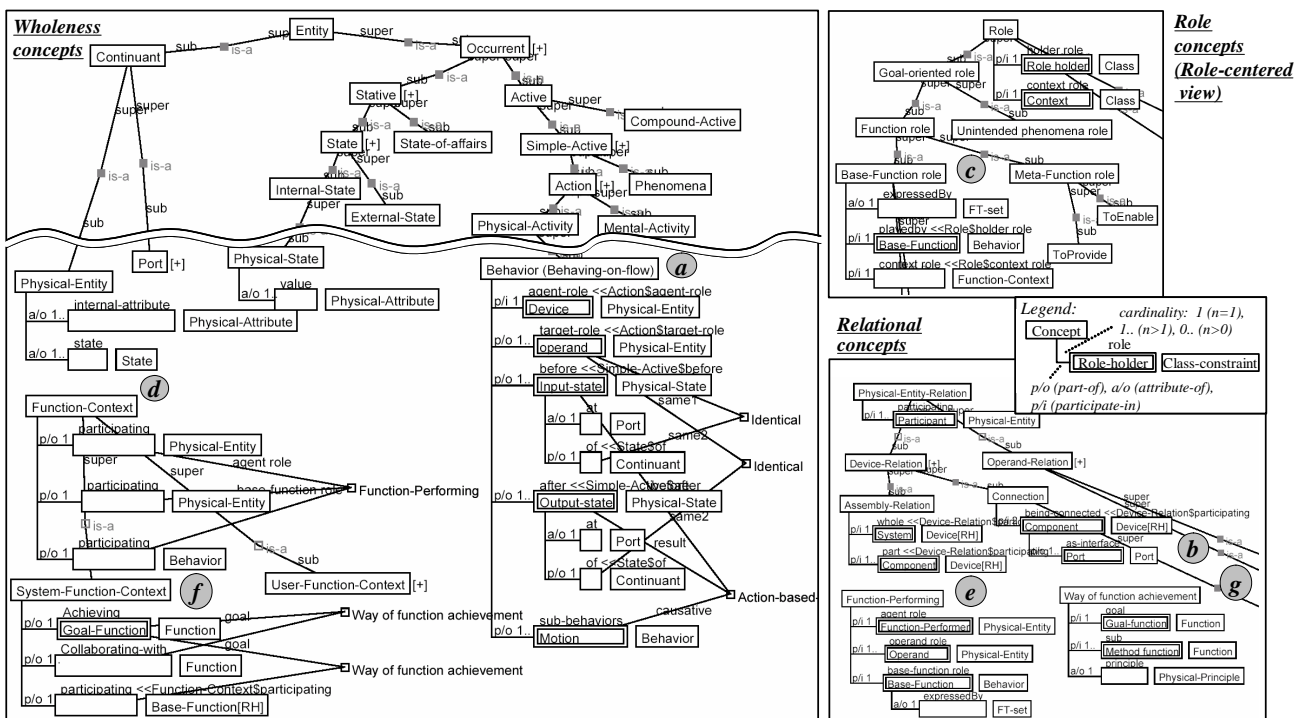


Fig. 2. A Portion of an Ontology of Device-centered Behavior and Function in Hozo.

is redefined as the role recognition of the behavior in the function context. The additional information for the role recognition of behavior can be described using *functional toppings (FTs)*, that is, *Obj-Focus*, *O-Focus*, *P-Focus* and *Necessity*. They represent information about such an *operand* that the designer intends to change (focus of intention). *Obj-Focus* specifies its kind such as substance or energy. *O-Focus* specifies the physical type of its physical attributes to change (such as temperature and phase). *P-Focus* specifies ports and represents focus on a flow of operand or medium. *Necessity* specifies the necessity of *operands* in the context. Such definition of function resolves the first issue and the second issue discussed in Section 3.

For the example of the heat exchanger mentioned in Section 3.1, its behavior is described as a thermal energy flow between two medium flows. The giving-heat function can be represented as; (1) *O-Focus* on temperature and (2) *P-Focus* on the medium flow which receives heat (heat destination). On the other hand, the removing-heat function can be represented as (1) the same *O-Focus*, (2) *P-Focus* on another source medium-flow which releases the heat energy and (3) the heat energy is not necessary (*Necessity*). Such functional toppings show the difference between these two functional interpretations. See our previous paper for detail of representation (Kitamura *et al.* 2002).

The values of the functional toppings are determined according to the function-context. The functional toppings (FTs) are localized representation of the surrounding context of the component. This locality contributes to obedience to the No-Function-In-Structure principle discussed in Section 3.1. Moreover, the values of FTs are limited within behaviors. It limits the space of teleological interpretation of a behavior. In fact, semi-automatic identification of functions can be done by enumerating possible all interpretations from given behavioral model (Kitamura *et al.*, 2002).

Moreover, in addition to the base-function, we identified *meta-function*. In comparison to a base-function which is a role of behavior (and device) for operand, a meta-function is a collaborative role played by a *function* for *another function* such as ToDrive and ToProvide. For example, ToDrive means that a function supplies energy driving another function. It is the result of teleological interpretation of causal relations among base-functions of different components. Each *base-function* has a specific type of function (called *function types*) as well, which represents

causal patterns of achievement for goals of each base-function of a component such as ToMake and ToMaintain (we redefined the ones in (Keuneke 1991)). For more detail on meta-functions, see (Kitamura *et al.*, 2002).

4.2 Functional Concept Ontology

We developed an ontology of generic functions (called functional concept ontology) (Kitamura *et al.*, 2002), which are sub-classes of the “function” class in the ontology of device and function. Figure 3 shows an overview of our modeling framework and a portion of the functional concept ontology.

It defines about 220 concepts in four kinds of *is-a* hierarchies with such operational definitions. For example, an energy function, “to shift energy”, is defined as a behavioral constraint: an existence of focused-energy flow between two different mediums and constraints which are satisfied functional toppings as a reflection of function context. It can be defined by the axioms inherited from the super-concept plus the following three axioms; (1) P-Focus on an inlet port and an outlet port, (2) Energy in the focused outlet port is made from energy in the focused inlet port, and (3) Mediums of the focused energies are different. *To take*, a subtype of *to shift* in the *is-a* hierarchy, is defined as FTs of *to shift* with an additional FT, P-Focus on the port of energy provider. Likewise, *to remove* is defined as that of *to take* with an additional FT, the energy taken is *unnecessary* as *Necessity* FT. For details, see (Kitamura *et al.* 2002).

4.3 Function decomposition tree

On the basis of the ontologies, a *function decomposition tree* is described as a functional model of a concrete artifact. According to the extended device ontology, the knowledge authors assign *roles* to physical entities in the target world. Functions of the function decomposition tree are instances of generic functions defined in the functional concept ontology. In the example of the coffee maker in Figure 3, the

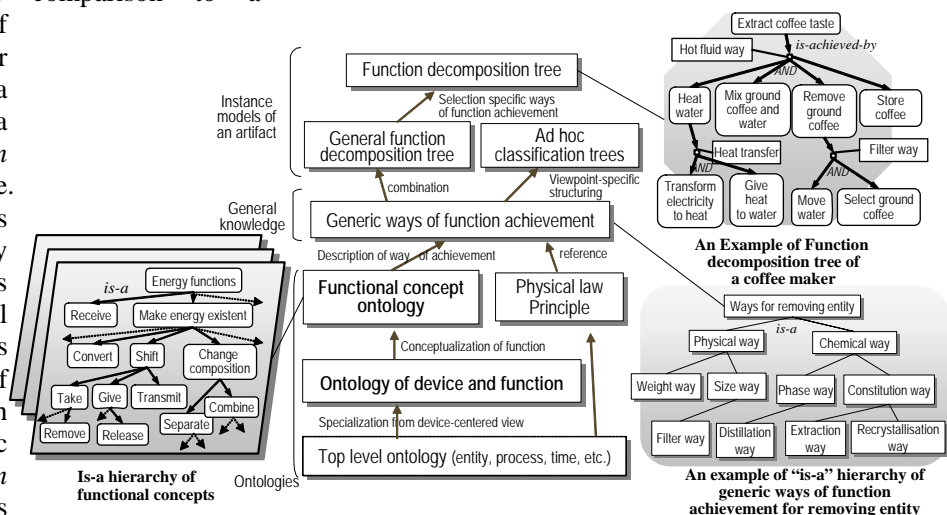


Fig. 3. Layered ontologies and functional knowledge.

goal function of the coffee maker is “to extract coffee taste”. The whole function is decomposed into finer functions, that is, “to heat water”, “to mix ground coffee and water”, “to remove ground coffee” etc. Its way of function achievement is conceptualized as “hot fluid way”, whose principle is extraction by heat.

A *general function decomposition tree* includes possible alternative ways of function achievement in OR relationship for a specific goal function. It shows *design alternatives* which are possible to be adapted or have been rejected in the previous design. It plays a crucial role in sharing design rationale.

4.4 Generic way of function achievement

The concrete *ways of function achievement* in function decomposition trees can be generalized into a *generic way of function achievement* (called functional way knowledge). Then, *ways* to achieve the same function are organized in *is-a* relations according to their principles (called an *is-a hierarchy of ways of function achievement*). We distinguish the organization as an *is-a* hierarchy from the other derivative organizations depending on viewpoints (called an *ad hoc classification tree*). This distinction is one of the solutions for the issue in Section 3.4. Figure 3 shows a portion of the “is-a” hierarchy of generic ways of function achievement for removing entity. They are categorized into “physical ways” and “chemical ways”, which are categorized into more specialized sub-classes. The filter way used in the coffee maker is sub-class of the “size way” of the “physical way”.

5. Deployment in Industry

The ontology and the modeling framework for functional knowledge have been deployed for over four years at the Plant and Production Systems Engineering Division of Sumitomo Electric Industries, Ltd. (hereinafter referred to as SEI) (Kitamura *et al.*, 2004). The purpose was to share functional knowledge among engineers about the production facilities used in their daily work. After a one-year study of our framework, test use was commenced in February 2001. In May, 2001, use on real problems encountered in daily work was started. The targets are manufacturing facilities that are mainly used in semiconductor manufacturing processes including machines to slice semiconductor ingots, machines to polish wafers, a tension control system, and inspection machines.

A knowledge management software named SOFAST® (abbreviation for Sumitomo Osaka-university Function Analysis and Systematization Tool and registered trademark of SEI.) has been deployed since December, 2002. It is designed to support the description of the functional decomposition trees and sharing in an intra-network. Using its client software, a user can describe function decomposition trees through a graphical user-interface and store them in its

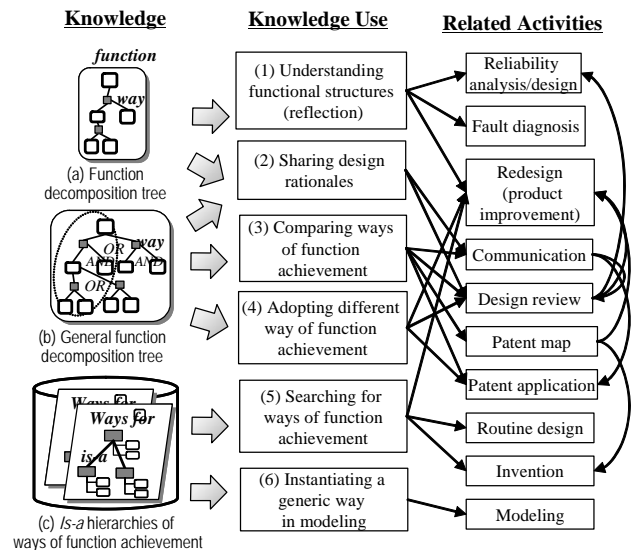


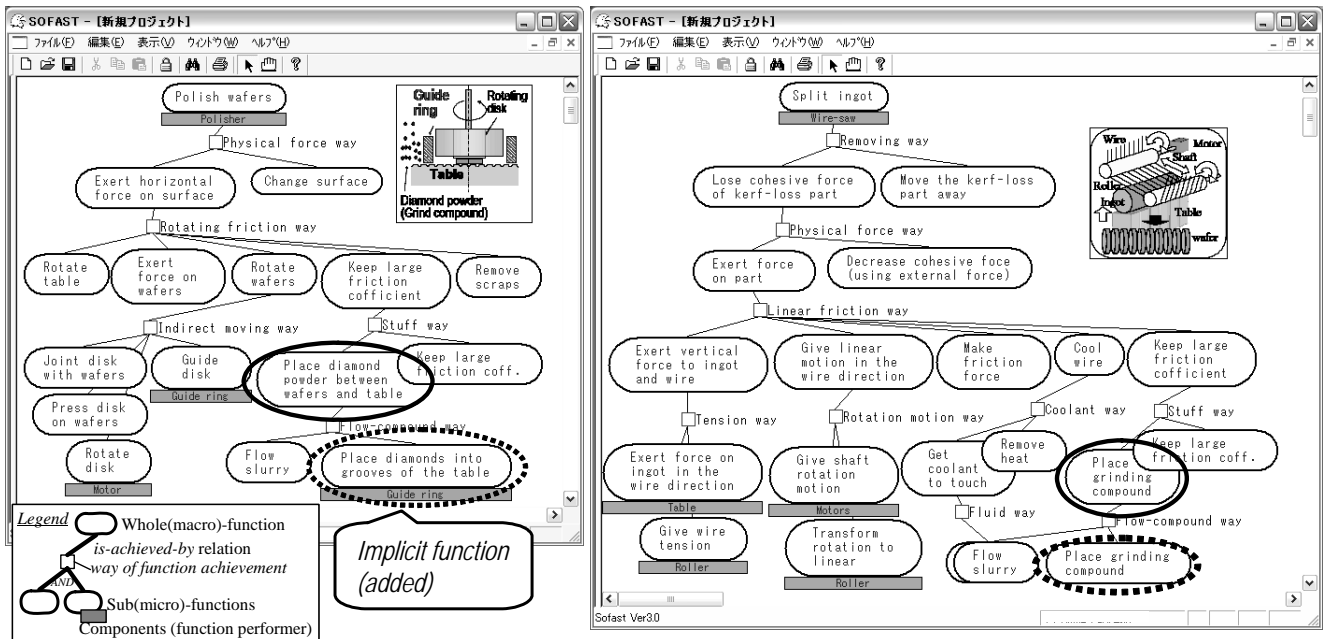
Fig. 4. Usages of Functional Knowledge

knowledge repository. Then, all users can search *ways of function achievement* in the repository to achieve the function of interest by specifying a goal function. A users' group of SOFAST software for companies was established in April, 2003.

5.1 Knowledge Usages in the deployment

Figure 4 summaries usages of the functional knowledge in the deployment. The left column enumerates kinds of functional knowledge discussed in the previous sections. Using each knowledge form, engineers (and/or a team of engineers) can do the engineer's activities shown in the center column. The right column shows the situations or the engineering tasks which can be facilitated by the activities. Some examples are discussed below.

One of the use of the (general) function decomposition tree ((a) or (b) in Fig. 4) is to clarify functional knowledge, which shows design rationale and used to be implicitly possessed by each engineer, and share it with other engineers (Usage (2) in Figure 4). The experiential evaluation by Sumitomo's engineers was unanimously positive. Such sharing was done for design-review where a design team does check a design and explore possible alternatives. As a replacement of a comparative table with a text, the engineers started to use the (general) function decomposition trees for design-review documents. After adopting our framework, the number of times the design reviews had to be done was successfully reduced to one third. This is because the function decomposition tree shows how to achieve the goal function as one of design requirements and then facilitates extensive discussion with other team members. Especially, the general function decomposition tree has strong effect on design review in order to compare alternative *ways* of achieving functions for each (sub) function (Usage (3) in Fig.4). It shows their features in comparison, and reasons for



Left: Fig. 5(a). Function decomposition tree of a polisher which is described for its improvement ,
Right: Fig. 5(b). Function decomposition tree of a wire-saw which is referred to.

adopting a specific way, or not, in the one figure. Such information is crucial in design review.

Writing a function decomposition tree according to the methodology gives designers the chance to reflect on good stimuli, which leads them to an in-depth understanding of the equipment (Usage (1) in Fig.4). Such a deep understanding of the equipment contributes to redesigning and solving problems with it.

For example, an engineer was not able to reduce the time a machine requires to polish semiconductor wafers after four months of investigation by adjusting the known working parameters. He consequently described its function decomposition tree shown in Figure 5(a), which is a screen snapshot of the SOFAST software⁴. He understood the guide ring has only one function, that is, to guide the movement of the rotating disk. The rotating disk freely moves inside the guide ring for polishing the wafer. Thus, he was not aware of another implicit function, “to place diamond powder between wafers and the table” (i.e., the function marked with dotted circle was missing). Then, he found it when he referred to that of the wire-saw shown in Figure 5(b). Although these two devices have different main functions and the wire-saw knowledge has been described by another engineer, he found the shared function “to keep a large friction coefficient” (marked with circles in Figures 5(a) and 5(b)) and its sub-function “to place grind compound” in the wire-saw model. As a result, he became aware of the missing function in the polisher model and its parameters for placing more diamond

powder (that is, the width of the guide ring) to obtain a high friction coefficient. Eventually, he reduced the necessary time to 76%, which was better than the initial goal. This improvement was achieved within three weeks. This example shows that functional knowledge tends to be implicit and awareness of such functionality gives engineers good stimuli for improvement. Moreover, reuse of functional knowledge between different facilities contributes to it.

Comparing design candidates as different ways to achieve functions (Usage (3) in Fig.4 of the general function decomposition tree) contributes to patent analysis and patent applications as well as design review mentioned above. In communications between engineers and patent attorneys in applying for a new patent, it is difficult to determine the product’s originality and to make appropriate claims. When the general function decomposition tree has been adopted as regular document format for patent application, the period was reduced to just one week from three or four weeks. The patent claims were increased and doubled in some cases, since the attorneys found extra differences with other patents by checking at each level of function decomposition. The same benefit was found by another company in the users’ group.

Generic knowledge about ways of function achievement ((c) in Fig. 4) helps designers search ways to achieve a function and/or alternatives in an existing product. In the deployment, a novice engineer developed an inspection machine in three days by systematically consulting generic ways of shedding light in the knowledge repository of SOFAST. Such development usually requires experts two weeks. Sharing *ways of function achievement* in SOFAST does not rely on the *is-a* hierarchy of generic ways but

⁴ The current version of SOFAST does not provide sufficient support for English. These snapshots are intended to show the functional decomposition trees for explanation and to provide rough images of SOFAST.

on searching for specific *ways* of function achievement by specifying the goal function. Nevertheless, discrimination of *is-a* relations from *ways of function achievement* has helped engineers avoid a great deal of confusion.

5.2 Effects of the Ontologies

One of the effects of the functional ontologies is to give prescriptive constraints for contents of functional knowledge. Because functional knowledge is intrinsically subjective rather than objective, without guideline, novice modelers would be puzzled in describing functional models. Especially, the extended device ontology provides users with hints on interpreting how a device works consistently. It provides concepts for assigning “roles” for each object in the target world. For the wire-saw example of difficulty discussed in Section 3.3, the wire can be considered as an *agent* (exerting force on ingots), an *operand* (moved by the roller) or a *conduit* (transmitting tension) depending on its location. According to semantic constraints in the extended device ontology, a possible way to consistently assign roles is to decompose the wire into two parts, a working wire playing an *agent role* and a transmitting wire playing both a *medium* role and a *conduit* role.

The explicit modeling of “way of function achievement” also helps modelers to eliminate the confusion between “what to achieve” and “how to achieve it”. A clear distinction between a general-specific hierarchy (*is-a* relations) and a whole-part hierarchy (*is-achieved-by* relations) helps knowledge authors to have consistent descriptions of function decomposition trees and *is-a* hierarchies of *ways of function achievement*. This avoids the confusion between the two, which has often occurred as discussed in Sections 2 and 3.4. For example, the example of “to weld” in Section 2 can be described as *fusion way* of the *joining function*. The fusion way has specific characteristics of the output that the operands are fused and they are hard to be separated. Although a functional concept “to join” loses some amount of information of “to weld”, what it loses goes to the characteristics of the fusion way. As a total, functional concepts are successfully made very generic without any loss of information.

Secondly, the functional ontologies provide an appropriate level of abstraction in order to recognize “commonality” between the products for sharing knowledge over different products. Usually, the engineering knowledge is described in a product-specific manner, which has restricted the applicability of knowledge to a specific product. Thus, engineers have difficulty to find related knowledge in different products or domains. The concept of function itself is a solution, since a function is intrinsically abstract and independent of its realization (e.g., structure and material). Nevertheless, many functions

in practice imply the way of function achievement such as “welding”, which is a kind of dependence on realization. It prevents to recognize the commonality between artifacts. In our framework, on the basis of the concept of the way of function achievement, we established an ontology of functional concepts, which enables the computer systems to find more artifacts to match with a specific artifact.

Lastly, our ontology-based modeling framework provides a knowledge medium to externalize engineers’ understanding which had been implicit. In the SEI’s deployment, engineers said that writing functional models of their own equipment gives them chances to reflect and obtain good stimuli, which leads them to an in-depth understanding of the equipment. The micro-macro hierarchy of the function decomposition tree enables the designer to systematically explore possible alternatives (for conceptual design) and/or causes of the problem (for problem solving) for each function. For example, fault tree analysis (FTA) for trouble shooting tends to make it difficult to enumerate all possible causes without a clear understanding of the function structure of the target device.

5.3 Lessons Learned

Although the deployment has been successfully done, we faced some difficulties in the deployment. This sub-section summarizes lessons learned, i.e., some issues for the successful deployment and some difficulties.

The first issue of the deployment was how to give a driving force to make engineers use new (unfamiliar) technology, i.e., the ontology-based modeling framework and the SOFAST software. Generally, engineers are very busy and have no strong motivation to use new technology in their daily work. Especially, in general situations of knowledge management projects, knowledge authors have no effective motivation to write their own knowledge and share it with others.

In the deployment, SEI’s managers established a regulation and a reward system for use of the modeling framework. The regulation in the department enforced engineer’s use of the general function decomposition tree as an official standard format of documents for design review. This regulation added no extra effort for engineers, because the function decomposition tree was adopted instead of conventional documents in a table form. This fact had a good effect for engineers from the viewpoint of the modeling cost. On the other hand, the rewards are given as a part of the company’s management system for evaluation of workers. Both organizational incentives successfully gave engineers a chance to write a function decomposition tree at least as a test use.

As well as such incentives, motivation of engineers themselves came from the direct benefit from writing functional models in contrast with the general situations of knowledge management mentioned above. In the deployment, the engineers say that they obtain benefits from writing functional models of their own equipment, since it gives them the chance to reflect and obtain good stimuli, which leads them to investigate possibilities in design or diagnosis systematically and exhaustively as discussed in Section 5.2. Moreover, sharing an initial successful use of a functional decomposition tree gave effective triggers to use by other engineers. The fact that we got a successful test-use in the early phase of the deployment played a crucial role in promoting our technology to other departments and other companies.

The second issue is how engineers learn the modeling framework. The ontological modeling framework needs training for writing functional models that are compliant with functional ontologies. Using the SOFAST software is easy to learn but writing function decomposition tree is not easy. Some engineers were puzzled out or wrote such functional decomposition models that are not compliant with the ontologies and thus not reusable for other products. In the deployment, the SOFAST users' group held many practices for writing functional models using real examples submitted from member-companies.

Moreover, it turned out that it is not easy for engineers to select appropriate functional concepts according to ontological definitions. People felt difficulty to learn definitions of the functional concepts. One of its reasons is that we give theoretically-sound names to the concepts, which are sometimes unfamiliar to engineers. Considering engineers' preference, terms (labels) to denote functional concepts could be a domain-specific vocabulary familiar to them. We never claim completeness of concepts in our functional concept ontology due to its nature and understand the necessity of extending it.

These difficulties found in the deployment gave us a motivation for the following twofold extensions of our ontological framework for easier use. One is treatment of lexical terms for representing functional concepts in order to reduce the difficulty in selection of functional concepts. The other is establishment of ontological guidelines in order to make ontological commitment easier.

6. Extension

In order to resolve some difficulties discussed in the previous section, we had developed a second version of SOFAST (the version number is 3. hereafter SOFAST V3). In this section, we discuss two extensions of SOFAST.

6.1. Lexical Layer

The experience of the deployment shows importance of flexible lexical terms for representing functional concepts. There could be a domain-specific vocabulary and different terms for the same concept. Thus, SOFAST V3 supports multi-layers of functional-terms shown in Figure 6. It consists of four layers, i.e., an ontology layer, a concept layer, a lexical layer, and a domain-specific layer. The *ontology layer* is for the functional concept ontology, which defines functional concepts with explicit relationship with behaviors. It is organized in a rather deep (from three to six levels) *is-a* hierarchy for each kind of target operands. Each functional concept has a term to represent the concept, which is just a label for human readability. We never claim the appropriateness of the labels themselves.

The *concept layer* is a simplified version of the organization of the functional concepts in the ontology layer, which is a single three-level hierarchy. Some intermediate concepts in the ontology layer are omitted in the conceptual layer. It makes engineers' understanding of the organization of the functional concepts easier. The concept layer is implemented in SOFAST V3 as *essential functional concepts*. The mapping between the concept layer and the ontology layer was made by the authors as a part of application development.

The *lexical layer* includes *usual functional words*, which are verbs for representing functions appearing in daily work. The usual functional terms have been prepared beforehand by collaboration with companies in the users' group. The domain specific layer has some vocabulary sets specific to domains or companies.

Such *usual functional words* are associated with *essential functional concepts*. A usual functional word can represent some essential functional concepts and vice versa. For example, a usual functional word "to put two objects together" can imply either of (at least) the following two functional concepts;

- (1) The two objects which are independent of each other become fixed together in a cohesive manner. This is a sub-concept of "to change

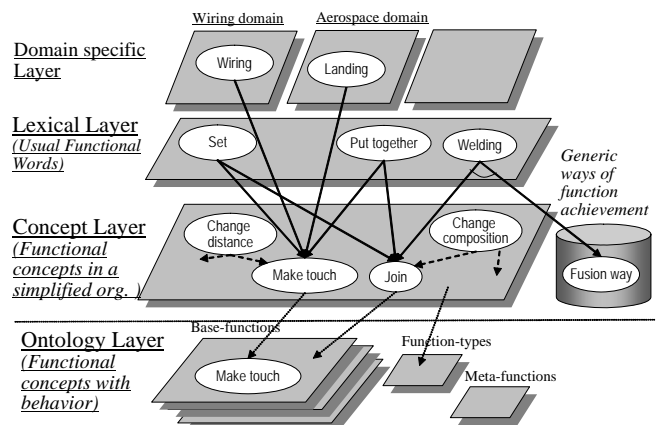


Fig.6. Multi-level Layers of Functional Terms with Examples

composition of operands”. In the concept layer, this concept is denoted by “to join”.

- (2) The distance between two objects changes from positive number to zero. This is a sub-concept of “to change distance”. In the concept layer, this concept is denoted by “to make operands touch”.

Thus, the usual functional word “to put together” has the mapping relations to the functional concepts “to join” and “make touch”. On the other hand, there are other usual functional words for these functional concepts. For example, “to make touch” can be referred to as “to set” as well.

Some usual function words imply not only functions but also specific ways of function achievement. For these words, the mapping shows a decomposition relation. For example, a term “to weld” can be decomposed into the “to join” functional concept and the fusion way.

The multi-layer functional terms are used in selecting a label for a function node in SOFAST V3. Users can select a function using either essential functional concepts or usual functional words. When a user selects a usual functional word which is associated with some essential functional concepts, the user has to select an essential functional concept from these candidates. Thus, a function node in the function decomposition tree has both a usual functional word and an essential functional concept. In the diagrammatic representation of the function decomposition tree in SOFAST V3, the usual functional word is used. When a user searches instances of functions in functional decomposition trees, the user can search using essential functional concepts. Thus, users can search functions independently of lexical (superficial) representation of the function.

6.2. Ontological Modelling Guidelines

In order to help easier commitment to the ontologies, the authors had established stepwise guidelines for describing functional knowledge shown in Table 1 (Kitamura & Mizoguchi, 2004b). They show ontological constraints for contents of models in a form of a check list in natural language. A model-author can check a function decomposition tree using the guidelines and then can modify it if necessary. The guidelines are categorized into three groups, i.e., Group F: guideline about functions and behavior, Group S: guidelines about relations between method-functions (i.e., sibling functions at a level of decomposition), and Group A: guidelines about “is-achieved-by” relations (between goal and method functions, i.e., decomposition relation) and ways of function achievement.

Table 1. Guidelines for function decomposition tree

F: About functions and behaviors

- F1.** A function represents “what to achieve” only and does not imply “how to achieve”.
- F1-1.** A device is a black-box. The inside is not shown at a level.
- F2.** A function represents (a teleological interpretation of) changes in physical things within the system boundary.
- F2-1.** Do not describe the designer’s activities.
- F2-2.** Distinguish product’s functions and manufacturing processes.
- F2-3.** Determine a system boundary with a pre- and post-process.
- F3.** Agent of functions should be a “device” in the physical world.
- F3-1.** A human operator can be regarded as a “device”.
- F3-2.** Designers and manufacturer should be distinguished.
- F3-3.** Sizes of devices decrease in function decomposition.
- F3-4.** A device can be virtual and dynamic.
- F4.** Decompose functions which imply kinds of operands and/or degrees of results for functions. Such implications are represented as attributes of ways of function achievement

S: About relations between sub-functions

- S1.** Identify states of operands that flow sub-functions.
- S2.** Time passes along this relation.
- S3.** Roles of things as operands should not be changed in a series of functions.

A: About “is-achieved-by” relation and way of function achievement

- A1.** The “is-achieve-by” relation represents aggregation
- A1-1.** The total changes in sub-functions should correspond to changes in the whole function.
- A1-2.** This relation does not imply a time interval.
- A1-3.** This relation is not an “is-a” relation.
- A2.** A sub-function should explicitly contribute to a macro-function.
- A2-1.** Explicate implicit sub-functions.
- A3.** The way of function achievement represents a single principle.
- A3-1.** Decompose compound principles
- A3-2.** Distinguish them from other ways at the principle level.
- A3-3.** If possible, conceptualize neither tools nor operands but principles
- A3-4.** A way should refer to a direct macro-function.
- A4.** Distinguish supplementary functions from essential functions.

Figure 7 shows an example of modification process of the function decomposition tree of the wire-saw which is shown in Figure 5(b). Figure 7 (a) shows the initial model and (b) the revised one. In the example, the whole function “to slice” in Figure 7 (a) is modified into “to split” shown in Figure 7 (b), because “to slice” implies “how to split” and provides specific information about the thinness of the split part. The former information is regarded as the way of function achievement. Thus, the functional term “to slice” does not follow the guideline F1 in Table 1. The goal of slicing here can be considered to be “to split a part from the target operand (i.e. ingot)”. It makes it possible to select other ways instead of “slicing” in the design. In reality, slicing with wire is not a simple way of function achievement but a composite as will be discussed later. The latter information (i.e., thinness), on the other hand, is regarded as the quantitative degree of a function. Thus, the function term “to slice” does not follow the guideline F4. Each way of function achievement has specific value of attributes like it. Then, such information can be used as conditions to select the way from all available ways of function achievement.

As we can see in Fig. 7(a), one might describe “to move table to wire” and “to move wire” as sub-functions. However, against A2 in Table 1, why these two sub(method)-functions can perform the

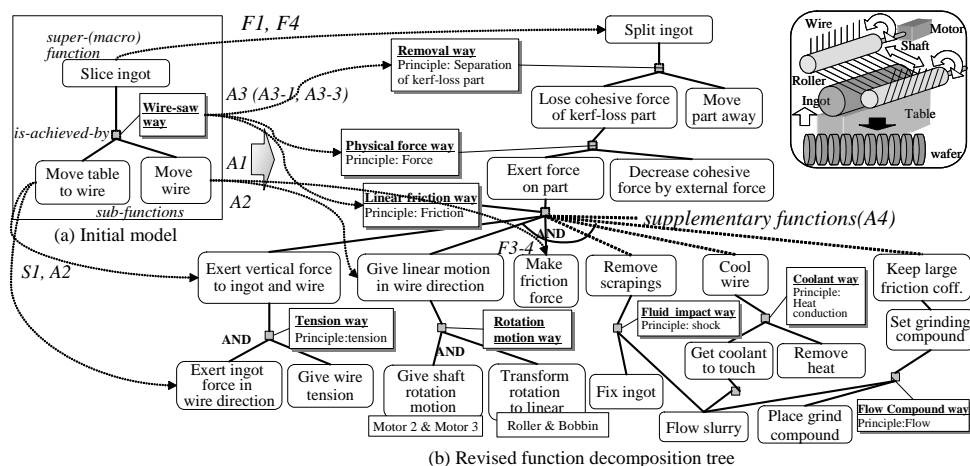
whole (goal) function is not clear. Moreover, against S2 in Table 1, it is unclear which operands flow between the two sub-functions. One reason is that there is a missing function, “to exert vertical force to ingot and wire”. The original sub-function “to move table to wire” contributes to the vertical-force sub-function.

Currently, the guidelines are only for human comprehension. Engineers themselves have to check their models according to the guidelines. Automatic checking of violations in the functional models based on the ontological guidelines is being investigated. Their definitions are structured with slots and constraints and include axioms as a result of deep insights into the behaviors and functions in physical systems. Such formal definitions can be used to automatically check the models.

Definitions and Categories of Function

readability, because some distinctions are independent from each other.

This cooling *environmental function* means physical changes of the system (called *physical environmental function*), while an *interpretational function* sets up one of necessary conditions of human's cognitive interpretation as shown in Fig. 8(b). For example, a clock has "to rotate hands (in the specific and constant rate)" as a *device function* and "to inform time" as an *interpretational function*, which requires human's cognitive interpretation. Similar concepts of functions can be found in the literature. Rosenman and Gero (1998) investigate *purpose* in *socio-cultural environment*. The situated FBS framework treats change of requirements (Gero & Kannengiesser, 2002). In our collaborative work with the Delft University of Technology, we are extending our framework to include user actions as well (van der Vegte *et al.*, 2004). The affordance-based design has been investigated in (Maier & Fadel, 2003).



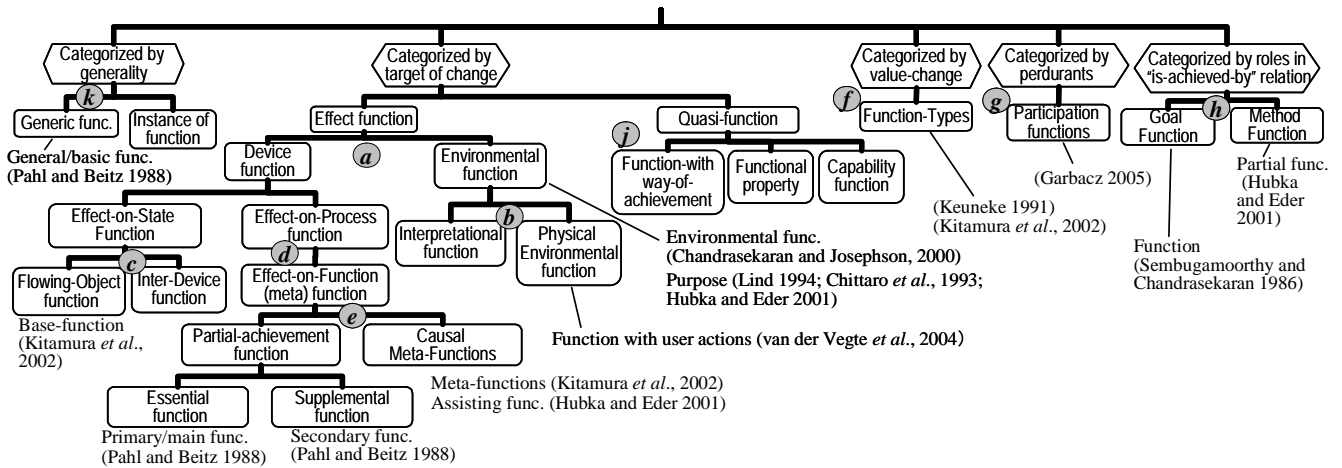


Figure 8. Some Other Definitions and Categories of Function in the Literature

which flow through the device (called *flowing object* function as shown in Fig. 8(c)). It can be generalized into *effect-on-state function* which means temporal changes of physical attributes. The effect-on-state function has another kind, that is, *inter-device* function which refers to changes of another device (called B-3 behavior in (Kitamura & Mizoguchi 2004a)). Its example is a rod's function "to push cam". The cam is another device, which is not considered as objects flowing through the rod.

On the other hand, the *effect-on-process function* represents effect on a process or its changes. Behavior as basis of function can be regarded as a kind of a process. Thus, as a subtype of the effect-on-process function as shown in Fig. 8(d), *effect-on-function function* (we can call meta-function) represents a role of a function for another function. It includes *partial-achievement function* and *causal-meta function* as shown in Fig. 8(e). The former is performed by a *method function* for a *goal function* in the *is-achieved-by* relation. The achievement function can be categorized into the essential function and the supplemental function. Distinction of primary function and secondary function (Pahl & Beitz 1988) is similar to this. The latter, causal-meta function, represents a role for another method function and is called a *meta-function* in Section 4.1 and (Kitamura *et al.*, 2002). Hubka and Eder (1988, 2001) categorize some types as assisting function.

The function-types have been identified by Keuneke (1991) based on patterns of value change as shown in Fig. 8(f). We redefined it as categorization of changes of values (Kitamura *et al.*, 2002) as discussed in Section 4.1. Garbacz (2005) defines *participation functions* according to categories of perdurants (i.e., occurrent in Figure 2. This is shown in Fig. 8(g)). They are "achieve", "accomplish", "maintain" and "process", which correspond to achievement, accomplishment, state and process, respectively.

In the context of the "is-achieved-by" relation, we can recognize a role of function, i.e., goal-function and

method-function as shown in Fig. 8(h). Function defined in (Sembugamoorthy & Chandrasekaran, 1986) is related with the former as discussed in Section 3. The latter is called partial function (Hubka & Eder 1988). In German systematic design methodology, the generic class of function is called general function or basic function (Pahl & Beitz 1988).

We recognize the following three kinds of *quasi-functions* as shown in Fig. 8(j). Although the authors do not consider them as kinds of function, it is found that a quasi-function is occasionally confused with a function. Firstly, a *function-with-way-of-achievement* implies a specific way of function achievement as well as a function. Its examples include washing, shearing, adhering (e.g., glue adheres A to B), "transportation by sea" (Hubka & Eder 1988), "link" (Hirtz *et al.*, 2002) as well as welding mentioned in Section 2. Because meaning of this type of function is impure, we regard this quasi-function.

Secondly, a *functional property*⁵ represents that an artifact (usually material) has a specific attribute-value which directly causes functionality. This is found in material science domain where a material whose function is dependent on its electronic, optical or magnetic property is called functional material (EPSRC, 2005). For example, if an electrical conductivity of a material is high (i.e., it has high conductivity property), the material can perform the "to transmit electricity" function. There is direct relationship between the high-conductivity property and the transmitting function. Lastly, a *capability function* represents that an entity can perform an activity which is not effect on others. For example, people say that "a human has walking function".

⁵ The term "functional" here is intended to represent neither mathematical dependence relation nor attributes of function but function-oriented property, though attributes of function include (not only) functional properties. Functional property is used as an antonym of mechanical or structural property.

Hubka and Elder (1988) also define a function as property of capability for achieving a goal. However, their definition implies effects on operands in contrast with the capability function here.

Generic tasks and the upper-level concepts

Our functional ontology is different from “task” knowledge of designing or diagnosing, which is activity of human or automated problem-solvers. If one ignores the difference between domain and task, the generic tasks and the generic methods (PSMs) in the task ontology research (e.g., (Schreiber *et al.*, 2000)) are similar to our generic functions (as the antonym of ‘instance of function’ as shown in Fig. 8(k)) and generic ways of function achievement, respectively. We focus on structuring knowledge about how to achieve functions (activities in domain world). We conceptualize the principle behind the sequence of activities (called method in both researches) as the way of function achievement. It helps us organize them in *is-a* hierarchies, though PSMs for a specific task are usually not organized well. Moreover, we distinguish function at the teleological level from behaviors at the objective level.

Behavior of artifacts is a kind of “process” by which we intuitively mean a sequence of state changes over time. We concentrate on physical process which represents temporal changes of physical quantities as we discussed in Section 4 and in this section. On generic “process”, extensive research has been done such as the process specification language (PSL) (ISO, 2003) for “activity” and its temporal relationships, formal ontologies for processes (e.g., that in (Sowa, 1995)), and the MIT process handbook for business activities (Herman & Malone, 2003) which taxonomy for business activities includes “activity-with-way” such as “buy in a store” like “welding” in Section 2.

In Fig. 2, we show some upper-level types such as ‘state’, though we aim not at a new proposal of the upper-level types but at clarification of our definition of the function-related types. Many upper-level ontologies include similar types. For example, in comparison with DOLCE (Masolo *et al.*, 2003), our types; *continuant*, *occurent*, *physical-entity*, *stative* (*occurent*) and *active* (*occurent*) in Fig. 2 roughly correspond to *endurant*, *perdurant*, *non-agentive physical entity*, *state* and *process* in DOLCE, respectively.

Our definition of function is based on the notation of role concept (Kozaki, et al., 2002, Sunagawa, et al., 2006). Much research has been conducted on representation of role; e.g., (Fan *et al.* 2001, Gangemi and Mika, 2003, Masolo *et al.* 2004, 2005, Sowa 1995). Our analysis of function as role discussed in Section 3 follows the role’s characteristics discussed in these papers. Both Masolo *et al.* and we introduce a context to define a role. The difference between the

two is that their context is a description of a situation on the basis of D&S theory (Gangemi and Mika, 2003) but ours is a normal concept defined in a base ontology like DOLCE. In this paper, we define two types of *contexts* for functions. Although we have our own ontology of description (representation in our terminology) and it is interesting to discuss how D&S and the ontology are different, this is beyond the scope of this paper.

The exclusive features of our theory of roles include that it clearly distinguishes between the instantiation of role itself and to play actions of a role and hence between the instance of a role and existence of the role playing thing (*role-holder* in our terminology). The ‘qua-individual’ in (Masolo *et al.* 2005) is similar to an instance of a *role-holder*. We conceptualize both a class of a role concept and a class-equivalent of a role-holder. Such distinction enables us to represent an unplayed role (i.e., an instance of role without instance of a role holder). For example, when a vacancy on a teacher post arises, there is no instance of the *teacher* role-holder but an instance of the *teacher role* which has properties such as subject. See details (Sunagawa *et al.* 2006).

Way of function achievement

Similarly to the *way of function achievement*, a feature of function decomposition can also be found as a “means” in (Malmqvist, 1997; Bracewell and Wallace, 2001; Wilhelms 2003). We defined *is-a* relations between conceptualized generic *ways of function achievement*, and investigated how to organize them.

The generic way of function achievement aims at its generality by pointing out partial (and abstract) information on structure and behavior. The similar knowledge in the literature, the design prototypes (Gero, 1990) and the FBS modeling framework (Umeda, *et al.*, 1996), include structural decomposition and physical features, respectively. In IDEAL (Bhatta & Goel, 1997), generic teleological mechanisms (GTM) are used (modified) to design different contexts based on analogies. In our approach, based on a limited set of functional concepts, designers can explore explicit *is-a* hierarchies of *ways of function achievement*.

TRIZ (TIPS) theory provides some patterns (or strategies) for inventions based on the contradiction between two physical quantities (Sushkov *et al.*, 1995). We did not concentrate on design strategies but on modelling schema. TRIZ theory also concentrates on physical principles (effects), although we established a clear relationship between physical principles and functional structures.

Limitations

Besides the production systems and facilities in the deployment, the ontologies have been applied to

modelling a power plant (Kitamura *et al.*, 2002), an oil refinery plant, a chemical plant, a washing machine, a printing device, and manufacturing processes. The models have taken into account changes in thermal energy, flow rate, and ingredients of fluids, including force and motion of operands. The current functional concept ontology can describe simple mechanical products, although it does not cover static force balancing and complex mechanical phenomena based on the shape of operands. The modeling framework currently cannot cope with the human's mental process, body movements (so-called therblig in Industrial Engineering), business processes, or software processes.

8. Concluding remarks

Ontological consideration on functionality of artifacts and extension of the ontology-based modeling methodology based on deployment have been discussed. The role of ontologies is to provide semantic guidelines to capture the target world consistently and controlled vocabulary for representation.

The authors and colleagues are currently investigating more simplified functional models as meta-data for web documents (Kitamura *et al.*, 2006). In this framework, a knowledge author can choose the level of description, from just a keyword representing the whole function of the device to a full functional model as discussed in this paper. Such functional knowledge associated with web documents in natural language will reduce the cost of modeling according to company's demand.

Moreover, aiming at interoperability between functional knowledge and failure knowledge, the authors and colleagues are extending the framework to cope with unintended behaviors such as faults (Koji, *et al.*, 2005).

Acknowledgements: The authors would like to thank Kouji Kozaki, Munehiko Sasajima, Eiichi Sunagawa, Sninya Tarumi, Naoya Washio for their contributions to this work. Special thanks go to Dr. Masayoshi Fuse, Mr. Masakazu Kashiwase, Mr. Shuji Shinoki and the engineers in the Plant and Production Systems Engineering Division of Sumitomo Electric Industries, Ltd. for their cooperation with the deployment. The authors are grateful to the two reviewers: Stefano Borgo and Tania Tudorache for their valuable comments.

Reference

- Bhatta, S. R., Goel, A.K. (1997). A Functional Theory of Design Patterns. In *Proc. of IJCAI-97* 294-300.
- Borst, P, Akkermans, H, Top, J. (1997). Engineering Ontologies. *Int'l Journal of Human-Computer Studies*, 46(2/3):365-406.
- Bracewell, R.H., Wallace, K.M. (2001). Designing a Representation to Support Function-Means based Synthesis of Mechanical Design Solutions. In *Proc of ICED 01*.
- Breuker, J., Hoekstra, R. (2004). Epistemology and ontology in core ontologies: FOLaw and LRI-Core, tow more ontologies for law, EKAW 2004 Workshop on Core Ontologies in Ontology Engineering, 15-27..
- Chandrasekaran, B., Goel, A.K., and Iwasaki, Y. (1993). Functional representation as design rationale. *Computer* 48-56.
- Chandrasekaran, B, Josephson, J.R. (2000). Function in Device Representation, *Engineering with Computers* 16(3/4):162-177.
- Chittaro, L., Guida, G., Tasso, C., Toppano, E. (1993). Functional and Teleological Knowledge in the Multi-Modeling Approach for Reasoning about Physical Systems: A Case Study in Diagnosis. *IEEE Transactions on Systems, Man, and Cybernetics* 23(6):1718-1751.
- Chittaro, L. & Kumar, A. N., (1997). Reasoning about function and its applications to engineering, *Artificial Intelligence in Engineering*, 12, 331-336.
- Cutkosky, M.R., *et al.* (1993). PACT: An Experiment in Integrating Concurrent Engineering Systems. *Computer*, January:28-37.
- De Kleer J. (1984). How Circuits Work. *Artificial Intelligence* 24:205-80.
- De Kleer, J. & Brown J.S. (1984). A Qualitative Physics based on Confluences. *Artificial Intelligence* 24:7-83.
- EPSRC (Engineering and Physical Sciences Research Council) (2005). <http://www.epsrc.ac.uk/ResearchFunding/Programmes/Materials/ResearchPortfolio/EngineeringFunctionalMaterials.htm>
- Fan, J., Barker, K., Porter, B., and Clark, P. (2001). Representing Roles and Purpose, in *Proc. of the international conference on Knowledge capture (K-Cap2001)*, 38-43.
- Gangemi, A., Mika, P. (2003) Understanding the Semantic Web through Descriptions and Situations, Int'l Conf. on Ontologies, Databases and Applications of SEMantics (ODBASE 2003), Catania, (Italy), November 3-7, 2003
- Garbacz, P. (2005). Towards a Standard Taxonomy of Artifact Functions, In *Proc. of FOMI 2005*.
- Gero, J.S. (1990). Design Prototypes: A Knowledge Representation Schema for Design. *AI Magazine* 11(4):26-36.
- Gero, J.S., Kannengiesser, U. (2002). The Situated Function-Behaviour-Structure Framework. In *Proc. of Artificial Intelligence in Design '02*, 89-104.
- Gruber, T. R. (1993). A translation approach to portable ontologies. *Knowledge Acquisition*, 5(2):199-220.
- Gruber, T, Olsen, G. (1994). Theory Component-Assemblies, Ontology Server. <http://www-ksl.stanford.edu>.
- 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*, 13:65-82.
- Herman, G. A., Malone, T. W. (2003). What is in the process handbook?, *Organizing Business Knowledge: The MIT Process Handbook*, MIT Press, 221-258.
- Hubka, V., Eder, W.E. (1988). *Theory of Technical Systems*. Berlin: Springer-Verlag.
- Hubka, V., Eder, W.E. (2001) Functions Revisited. In *Proc. of ICED 01*.
- ISO TC184/SC4/JWG8 (2003). Process Specification Language, Part 1, 11 and 12, [http://www.tc184-sc4.org/SC4_Open/SC4_Work_Products_Documents/PSL_\(18629\)/](http://www.tc184-sc4.org/SC4_Open/SC4_Work_Products_Documents/PSL_(18629)/).
- Keuneke ,A.M. (1991). A. Device Representation: the Significance of Functional Knowledge. *IEEE Expert* 24:22-25.

- Kitamura Y, Sano T, Namba K, Mizoguchi R. (2002) A Functional Concept Ontology and Its Application to Automatic Identification of Functional Structures. *Advanced Engineering Informatics* 16(2):145-163.
- Kitamura Y, Mizoguchi R. (2003) Ontology-based description of functional design knowledge and its use in a functional way server. *Expert Systems with Application* 24(2):153-166.
- Kitamura Y, Mizoguchi R. (2004a) Ontology-based systematization of functional knowledge. *Journal of Engineering Design*, 15(4): 327-351.
- Kitamura, Y., Mizoguchi, R., (2004b) Ontology-based Functional-Knowledge Modeling Methodology and its Deployment, *Proc. of EKAW 2004*, 99-115.
- Kitamura, Y., Kashiwase, M. Fuse, M., Mizoguchi, R. (2004) Deployment of an Ontological Framework of Functional Design Knowledge, *Advanced Engineering Informatics*, 18(2):115-127.
- Kitamura, Y., Washio, N., Koji, Y., Sasajima, M., Takafuji, S., & Mizoguchi, R. (2006) An Ontology-Based Annotation Framework for Representing the Functionality of Engineering Devices, In *Proc. of ASME IDETC/CIE 2006*, DETC2006-99131, To appear.
- Koji, Y., Kitamura, Y. & Mizoguchi, R. (2005). Ontology-based Transformation from an Extended Functional Model to FMEA, In *Proc. of the 15th International Conference on Engineering Design (ICED '05)*, CD-ROM, 264.81 (Abstract: pp.323-324).
- Kozaki, K., Kitamura, Y., Ikeda, M., Mizoguchi, R. (2002). Hozo: An Environment for Building/Using Ontologies Based on a Fundamental Consideration of "Role" and "Relationship", *Proc. of EKAW2002*, 213-218, 2002
- Lee, J. (1997). Design rationale systems: understanding the issues. *IEEE Expert*, 12(3):78-85.
- Lind, M. (1994). Modeling Goals and Functions of Complex Industrial Plants. *Applied artificial intelligence* 8:259-283.
- López, M. F., et al. (1999). Building a Chemical Ontology using Methontology and the Ontology Design Environment, *IEEE Intelligent Systems*, Jan./Feb., 37-46.
- Maier, J. R. A., Fadel, G. M. (2003). Affordance-based methods for Design, *Proc. of DETC 03*, DTM-48673.
- Malmqvist J. (1997). Improved function-means trees by inclusion of design history information. *Journal of Engineering Design* 8(2):107-117.
- Masolo, C., Borgo, S., Gangemi, A., Guarino, N., Oltramari, A. (2003), WonderWeb Deliverable D18, Ontology Library, IST Project 2001-33052 WonderWeb: Ontology Infrastructure for the Semantic Web
- Masolo, C., Vieu, L., Bottazzi, E., Catenacci, C., Ferrario, R., Gengami, A., and Guarino, N. (2004). Social Roles and their Descriptions, in *Proc. of the 9th Int'l Conf. on the Principles of Knowledge Representation and Reasoning (KR2004)*, 267-277.
- Masolo, C.; Guizzardi, G.; Vieu, L.; Bottazzi, E.; Ferrario, R. (2005) Relational Roles and Qua-individuals, In *Proc. of the AAAI Fall Symposium on Roles, an interdisciplinary perspective* November 3-6, 2005.
- Mizoguchi, R., et al. (2000). Construction and Deployment of a Plant Ontology, *Proc. of EKAW 2000*, 113-128.
- Mizoguchi, R. (2005). The Role of Ontological Engineering for AIED Research, The international journal published by ComSIS Consortium, 2(1), <http://comsis.fon.bg.ac.yu/ComSIS/Vol2No1/InvitedPapers/RMizoguci.htm>
- Miles, L.D. (1961). *Techniques of value analysis and engineering*. McGraw-hill.
- Pahl, G., Beitz, W. (1988). *Engineering design - a systematic approach*. The Design Council.
- Rosenman, M.A., Gero, J. S. (1998). Purpose and function in design: from the socio-cultural to the techno-physical, *Design Studies*, 19, 161-186.
- Sasajima, M., Kitamura, Y., Ikeda, M., Mizoguchi, R. (1995). FBRL: A Function and Behavior Representation Language. *Proc. of IJCAI-95*, 1830-6.
- Schreiber, G., Akkermans, H., Anjewierden, A., de Hoog, R., Shadbolt, N., Van de Velde, W. and Wielinga, B. (2000). Knowledge Engineering and Management - The Common-KADS Methodology, The MIT Press, Cambridge, MA.
- Sembugamoorthy, V, Chandrasekaran, B. (1986). Functional representation of devices and compilation of diagnostic problem-solving systems. In *Experience, memory and Reasoning* 47-73.
- Sowa, J. F. (1995). Top-level ontological categories, *International Journal of Human-Computer Studies*, 43(5/6) 669-685.
- Stone, R. B., Chakrabarti, A. (eds.) (2005) Special Issues: Engineering applications of representations of function, *AI EDAM*, 19(2 and 3).
- Sunagawa, E., Kozaki, K., Kitamura, Y., Mizoguchi, R. (2006). Role Organization Model in Hozo. In *Proc. of the EKAW 2006*, To appear.
- Sushkov, V.V., Mars, N.J.I., Wognum, P.M (1995). Introduction to TIPS: a Theory for Creative Design. *Artificial Intelligence in Engineering* 9.
- 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* 10:275-288.
- Umeda Y, Tomiyama T. (1997). Functional Reasoning in Design, *IEEE Exert*, March/April:42-8.
- van der Vegte WF, Kitamura Y, Koji, Y., Mizoguchi R. (2004). Coping with Unintended Behavior of Users and Products: Ontological Modeling of Product Functionality and Use, In *Proc. of CIE 2004: ASME 2004 Design Engineering Technical Conferences and Computers in Engineering Conference 2004*. DETC2004-57720.
- Vermaas, P. E. (2005). Promises of a Philosophical Analysis of Technical Functions, In *Proc. of FOMI 2005*.
- West, M. (2004). Some Industrial Experiences in the Development and Use of Ontologies, *EKAW 2004 Workshop on Core Ontologies in Ontology Engineering*, 1-14.
- Wilhelms S. (2003). A Conceptual Design Support System using Principle Solution Elements. *Proc. of ICED 03*.
- Yoshioka, M., Umeda, Y., Takeda, H., Shimomura, Y., Nomaguchi, Y., Tomiyama, T. (2004). Physical Concept Ontology for the Knowledge Intensive Engineering Framework. *Advanced Engineering Informatics*, 18(2), 95-113