In New Frontiers in Artificial Intelligence: Joint JSAI 2005 Workshop Post-Proceedings, LNCS 4012, Springer-Verlag Berlin / Heidelberg, pp. 17-28, June, 2006.

Towards Ontologies of Functionality and Semantic Annotation for Technical Knowledge Management

Yoshinobu Kitamura, Naoya Washio, Yusuke Koji, Riichiro Mizoguchi

The Institute of Scientific and Industrial Research, Osaka University
8-1, Mihogaka, Ibaraki, Osaka, Japan
{kita, washio, koji, miz}@ei.sanken.osaka-u.ac.jp
http://www.ei.sanken.osaka-u.ac.jp/

Abstract. This research aims at promoting sharing of knowledge about functionality of artifacts among engineers, which tends to be implicit in practice. In order to provide a conceptual viewpoint for modeling and a controlled vocabulary, we have developed an ontological framework of functional knowledge. This framework has been successfully deployed in a manufacturing company. This paper firstly discusses an ontological definition of the concept of function from a device-centered viewpoint. Then, other types of function are discussed in order to place our definition in the related concepts in the literature. Next, as an application of the ontologies, we propose a metadata schema based on the functional ontologies for functional annotation in the Semantic Web. The functional metadata annotated to technical documents show designers intentions behind the documents and contribute to efficient retrieval of the documents. Moreover, task-oriented transformation and interoperability with other schemata can be realized based on the ontologies.

1 Introduction

The recent situation in engineering requires effective sharing of product knowledge among engineers. As well as data-level knowledge such as design drawings, geometry data in CAD systems and values of physical quantities, knowledge about functionality is very important to be shared. Intuitively, a function of a product explains what users can get using it (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 in so-called function structure (i.e., "how things work"). Such functional knowledge shows a part of designer's intention of artifacts (so-called design rationale (DR)) [2,16,26].

Nevertheless, in the current practical situation, such knowledge tends to be implicit behind the data-level knowledge. Even if such knowledge is explicitly described, it scatters around documents in natural language in an ad hoc manner. Its retrieval relies mainly on keyword-based search. Then, few such technical documents have been efficiently reused. For example, one might describe "to weld metals" as a function of a welding machine in a "verb+noun" style in Value Engineering [20]. 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 operation 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 fusion. If a function of the welding machine is described as "to join", the commonality between two facilities can be found and then a search engine can find them. This issue, that is, distinguishing "what to achieve" from "how to achieve", is not a terminological but ontological.

The goal of this research is to add semantic annotation of functional knowledge based on an ontology for such technical documents in the Semantic Web. A functional annotation for a document shows functionality of a device mentioned in the document, what components achieve it and/or how to achieve it (function structure). It shows designer's intention behind the design drawings or semantic information for natural language documents. The semantic information enables us to search (and integrate) documents using a controlled vocabulary and relationship among concepts. A metadata schema for functional knowledge is designed to provide fundamental concepts such as function and entity, properties (relations) among functions and controlled vocabulary for generic functions. The fundamental concepts help knowledge authors to describe annotation 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.

Although much research has been conducted on the representation of function in Artificial Intelligence [2,4,8,17,25], engineering design [5,6,18,21] and Value Engineering [20], there is no common definition of the concept of function itself [2,6,26] and semantic constraints are not enough for deriving effective guidelines for consistent annotation. The authors have established an ontological modeling framework for functional knowledge [9-12]. This framework includes an ontology of device and function as conceptual viewpoint and a functional concept ontology as a controlled vocabulary. This framework has been successfully deployed in a manufacturing company in Japan for sharing functional knowledge [11]. These ontologies form a basis of the metadata schema.

This paper discusses an ontological definition of function and its application to a metadata schema in the Semantic Web. Section 2 presents the ontologies about functions, which are defined on the basis of the concept of "role" in Ontological Engineering. It gives more detailed definitions to our previous definition in [9,10]. Section 3 discusses on other types of function in order to place our definition in the functional world in the literature. On the basis of the ontologies in Section 2, Section 4 proposes a metadata schema for functional annotation. The discussion in Section 3 contributes to realization of interoperability with other terminologies (i.e., schemata) for generic functions. Then, related work is discussed followed by some concluding remarks.

2. Ontological Definition of Functionality

Figure 1 shows a portion of ontological definitions of function-related concepts in our ontology editor of an environment for building/using ontology named Hozo [15].

Concepts are represented as frames (denoted by nodes in Fig. 1) 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 relation 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 relation concept has slots of participant concepts (*participate-in* relation. denoted by "p/i") and the attribute-slots.

The key concept of our definition of functionality is the "role" concept in Ontological Engineering. Intuitively, a role is something that can be played by an entity in a context. Precisely, in [23], a role is the secondness concept which is dependent on a pattern of relationship. In [19], 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 [15]. We distinguish role (something to be played) from role-holder (something playing (holding) a specific role). For example, a man (class constraint for role) can play "husband role" (role concept) in a "marriage" relation (role context), who is called "husband" (role holder). Using Hozo, the marriage relation has two slots with *participate-in* relation, one of which is defined as a husband role with a man as a class constraint. It is defined also in the "married

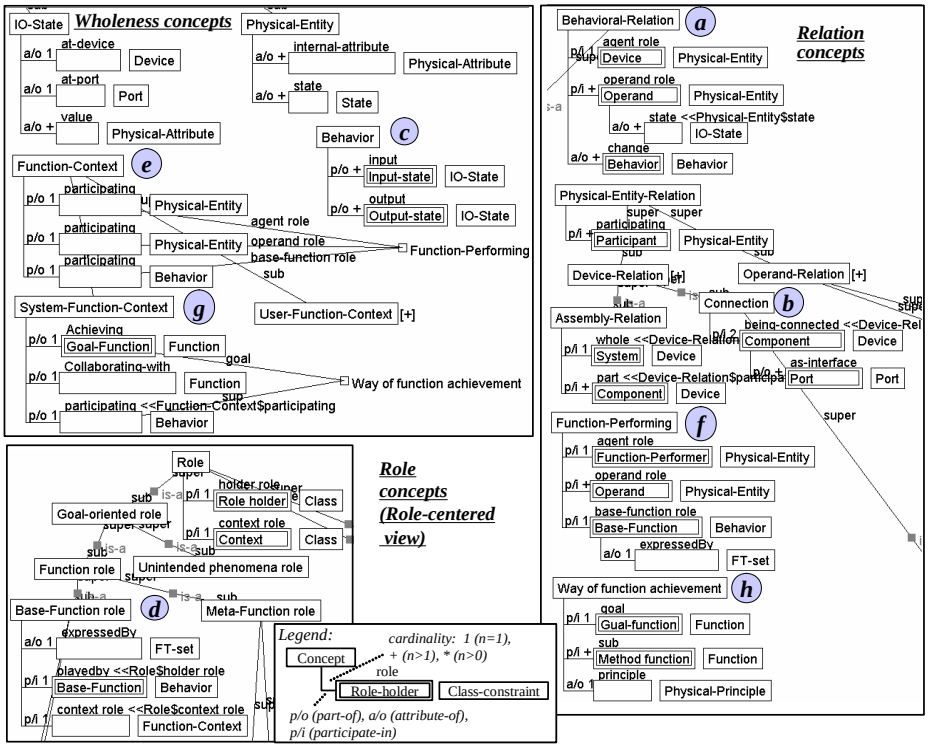


Fig. 1. A portion of an ontology of device-centered behavior and function in Hozo

couple" which is a wholeness concept corresponding to the "marriage relation".

As a basis of modeling of functionality, we adopt the device-centered view and then define the concepts of *device* and *behavior* (italics are defined terms). The *device* concept is defined as a role-holder in *behavioral-relation* between two *physical-entities* (Fig. 1 (a)). One of them plays the *agent* role, which is called *device*. It operates on the other entity (*operand* which is another role-holder) and changes its *physical-attributes*. Each *device* is connected to each other through its input and output *ports* (Fig. 1 (b)). The *operand* is something flows through the *device* and is affected by the *device*. The *operand* role can be played by fluid, energy, motion, force, or information. It has *IO-States*, which represents values of *physical-attributes* at a *port* of a *device*. The pairs of *IO-States* at *input ports* of a *device* and those at *output ports* of the same *device* are defined as *behavior* (Fig. 1 (c)). It represents objective (i.e., independent of the context) conceptualization of its input-output relation as a black box. Note that such a *behavior* of a *device* is founded [19], since a *behavior* of a *device* affects an *operand* and causes its temporal changes.

Such a device-oriented view comes from system dynamics theory and German systematic design approach [21], which is called a device ontology. We extended them by redefining the concepts of *behavior*, *conduit*, and *medium* [10]. We categorized the meanings of behavior into four types (from B0 to B3), one of which, called *B1-behaviour*, corresponds to the definition of *behavior* mentioned above.

In comparison with *behavior*, *function* is related to intention of a designer or a user (i.e., teleological). According to the definition of role in [19] and ontological consideration in [12], a function is a role. Firstly, a function is anti-rigid [19] and context-dependent (dynamic), because a function of a specific device can be changed without losing the device's identity. 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 flows from the warmer fluid to the cooler fluid. 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, i.e., a context. Moreover, a function of a system can be recognized according to a user's goal (e.g., a chair can be used as a ladder or a hammer). A function can be performed by different components. A component can perform multiple functions simultaneously. Secondly, a function is founded [19] as *behavior* does. For example, the definition of the removing-heat function of a radiator refers to the decrease of temperature of the warmer fluid (i.e., external entity) as input and output.

Thus, a (base-)*function* is defined as a role concept which is played by a *behavior* under a *function-context* (Fig. 1 (d)). In the *function-context*, there is a *function-performing* relation among two *physical entities* and a *behavior* (Fig. 1 (e)). In the relation, a *behavior* plays a *base-function* role, which is called a *base-function* role holder (Fig. 1 (f)). A *device* which performs the *behavior* 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 *function-context* represents teleological goals to be achieved by the *function*. A function-context of a function of a component in a system (called *System-Function-Context*, Fig. 1 (g)) can be determined by a goal function and method (sibling) func-

tions, which are defined in *way of function achievement* relation (Fig. 1 (h)). It means that the *goal function* can be achieved by a sequence of functions as *method-functions*. This is similar to the function decomposition in the German-style systematic design methodology [21], whole-part relation [17], and "degree of complexity" [6]. However, the basis of the function-achievement such as a physical principle is explicated as a *way-of-function-achievement*. It forms a function decomposition tree, which represents how to achieve a goal function by components as a designer's intention.

These definitions give more detailed definitions of our definition in previous papers [9,10], that is, a function as a teleological interpretation of a B1-behavior under a goal. The "interpretation" is defined as interpretation of a role of a behavior in a function context as a "goal". The teleological interpretation of behavior to function can be described using functional toppings (FTs), which are primitives of additional information to behaviors, that is, Obj-Focus, O-Focus, P-Focus and Necessity [9]. 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 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.

We developed an ontology of generic functions (called functional concept ontology) [9], which are sub-classes of the *function* class (its portion will be shown in Fig. 3). For example, an energy function, "to shift energy", is 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 the *shifting* function in the *is-a* hierarchy, is defined with an additional FT, P-Focus on the port of energy provider. Likewise, "to remove" is defined as that of the *taking* function with an additional FT, the energy taken is unnecessary as Necessity FT. On the other hand, "to give" concept can be define as P-Focus on another medium-flow receiving heat (heat destination). Such functional toppings show the difference between these two functional interpretations of the heat exchanger mentioned above.

3. Other Kinds of Function

The definition of the concept of function in Section 2 is done strictly from the device-centered viewpoint, which is intended to prescribe guidelines to functional modeling. Other types of function, however, still remain to be investigated. In order to place our definition of function in other definitions of functions, this section discusses rather descriptive definitions of other kinds of function as shown in Fig. 2 on the basis of the discussion in [12]. They represent viewpoints (or context) for human's perception of a function. Thus, a device can achieve some functions in different categories simultaneously. Note that Fig. 2 shows an *is-a* hierarchy only for readability, because some distinctions are independent from each other.

Firstly, the function discussed in Section 2 represents changes of entities (behaviors) within the system boundary (here we call *device function*). On the other hand, an

environmental function includes changes outside of the system boundary, especially, those related to users or user actions. For example, an electric fan performs moving-air function as a device function and cooling function for human body as an environmental function, where the cool-down effect by wind is on human body and thus outside of the system boundary. 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. 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.

Chandrasekaran and Josephson discuss a similar kind of function called environment function as effect on environment (the surrounding world of the device) [2]. Some researchers distinguish purpose from function (e.g., [17]), where the purpose represents human-intended goal in the similar sense to this *environmental function* or *interpretational function*. Hubuka distinguishes the purpose function as effects from the technical function as internal structure [6]. The situated FBS framework treats change of requirements [4]. In our collaborative work with the Delft University of Technology, we are extending our framework to include user actions as well [27].

Secondly, the *base-function* discussed in Section 2 refers to temporal changes of physical attributes of objects which flow through the device (called *flowing object function* here). 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 [10]). 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*, *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*. The former is performed by a method function for a goal function in the *is-achieved-by* relation. The latter represents a role for another method function and is called a meta-function in [9].

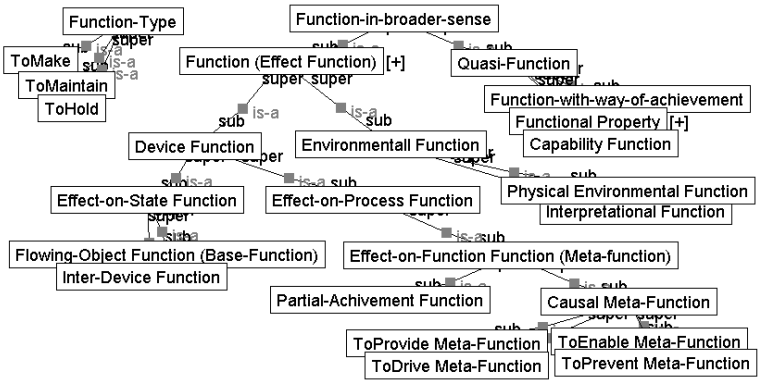


Fig. 2. Descriptive categories of function

Thirdly, the function-types are additional descriptors for the functions discussed thus far. They represent causal patterns of achievement for goals of each function of a component such as ToMake and ToMaintain (we redefined the ones in [8]) [9].

Fourthly, we recognize the following three kinds of *quasi-functions*. Although the authors do not consider them as kinds of function, it is found that a *quasi-function* is 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) as well as welding mentioned in Introduction. Because meaning of this type of function is impure, we regard this *quasi-function*. Secondly, a *functional property* 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. 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".

4. *Funnotation* Metadata Schema for the Semantic Web

In the semantic web context, our ontology can be used as a metadata schema for engineering documents as shown in Fig. 3. It enables us to describe metadata representing functionality of engineering devices mentioned in documents. Such metadata can be regarded as "content descriptors" like keywords or "logical structure" of "content representation" like a summary or an abstract in terms of categorization in [3]. By the logical structure, we here mean the relationship among functions such as functional decomposition. Functional metadata explicates the design rationale underlying design documents such as design drawing.

The proposed metadata schema, called *Funnotation* Schema hereafter, has been built intended to annotate web resources about artifacts from the functional aspects. The schema consists of layers (sub-schemata), that is, F-Core, B-Unintended, F-Vocab and F-Ways schemata as shown in Fig. 3. The schema is represented in OWL [29]. F-Core schema defines fundamental concepts based on the functional ontology discussed in Section 2. Some of OWL classes and properties in F-Core are shown in Table 1. For example, the *agent* property denotes that an *entity* can perform *function* as an agent. The *part_function* property is used for representation of functional decomposition trees. Verbs such as "*convey*" and "*separate*" are defined in F-Vocab schema as subclasses of the class of *function*. Those terms come from the functional concept ontology discussed in Section 2. F-Ways schema defines generic function-achievement ways, which are generalized from the concrete ways of function achievement in the function decomposition trees. The definition of each way of function-achievement is composed of the principle on which the achievement is based, the goal function and sub-functions which collectively constitute the way of function achievement.

The *Funnotation* schema enables users to represent functional metadata (called *Funnotation* metadata) with RDF [30] which include (1)Functions of the device/component/part of interest, (2)Function-achievement ways used, (3)Function decomposition trees representing the functional structure of the device, and (4)Generic function decomposition trees including alternative ways of function achievement. While (3) and (4) correspond to a full model of the functional structure, (1) and (2) to "indexing" information (content descriptors) representing some portion of the full model. Figure 4 shows an example of functional metadata for an explanation of a wire saw. It shows which represent that the wire saw is an instance of the *device* (*Funnotation:device*) and that it performs an instance of the *splitting function* (*Funnotation:split_entity*) as shown in the *Funnotation:has_function* property (which is the inverse property of the *Funnotaion:agent* property).

Such metadata enable us to search documents of engineering devices using their functions using a common vocabulary and *is-a* hierarchies of functional concepts. Here, the discrimination between functions and ways plays an important role. As mentioned in the introduction, usually both concepts are confused and thus it causes failure of search by functions. As well as the metadata by functions, secondly, one can describe ways of function achievement used in the device. In fact, Fig. 4 shows that the instance of *funnotation:frictional_way* is linked to the *split_entity function* instance via *selected_way* property to demonstrate the wire saw achieves its main function using *frictional_way*. Thanks to our functional ontologies, a user can search functions and ways separately.

Furthermore, by adding metadata about sub-functions to the metadata shown in Fig. 4, one can describe a function decomposition tree of a device as metadata of a document about the device. Many design documents describe only results of design activities without design rationale. The functional decomposition tree as metadata gives a

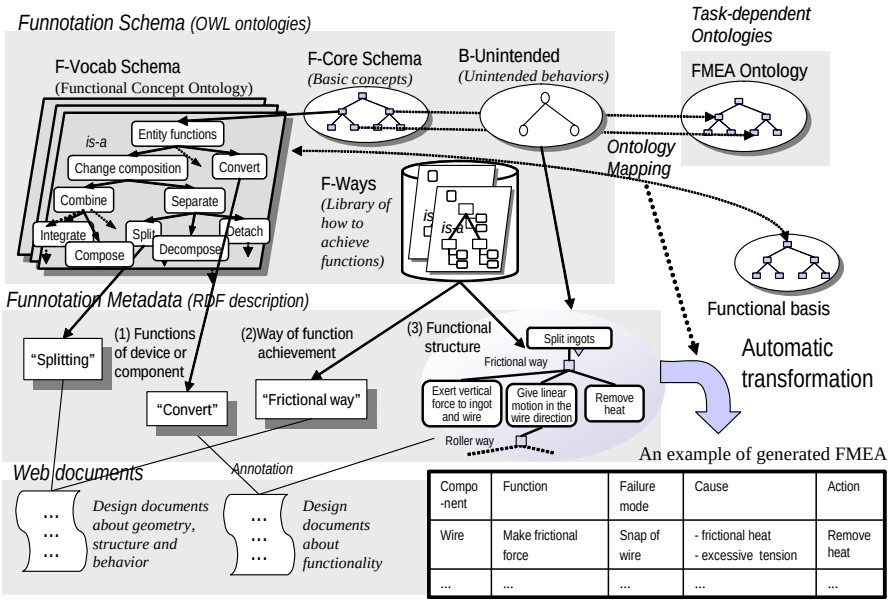


Fig. 3. *Funnotation* framework

part of the design rationale of devices described in the document. For few documents describing functional structures, the functional decomposition tree gives a kind of a summary or an abstract of the document.

Such functional metadata can play a crucial role in knowledge sharing among engineers in practice. In fact, in the deployment of our ontological framework in a manufacturing company [11], the functional modeling framework helps engineers share designer's intentions in engineering teams for design review and patent application as knowledge media. Moreover, it contributes to solving engineering tasks such as trouble-shooting and redesign [11]. Although the deployment has been done in a conventional client-server system, the same effect can be expected for that of the Semantic Web version.

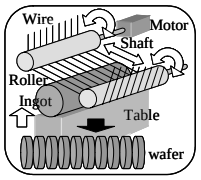
Furthermore, the metadata can be automatically transformed into the form user's (engineer's) task requires according to the ontology mapping. In the current implementation, a knowledge transformation system can generates FMEA (Failure Mode and Effects Analysis) sheets for reliability analysis by transforming an extended functional model [13] as shown in Fig. 3. The transformation is done by referring to the ontology mapping knowledge between ontologies of the both knowledge models: the extended function model and the FMEA model. The *Funnotation* schema has a layer

Table 1. Classes and properties of F-Core (portion)

Class			
entity	Physical entity		
function	Interpretation of behavior under a goal		
way	Way of function achievement: conceptualization of the principle essential to the achievement of the parent (goal) function by the sub(part)-functions		
Property			
Name	**Domain**	**Range**	
agent	*function*	*entity*	*Function* is achieved (performed) by the *entity*
part_function	*function*	*function*	*Function* in the Domain (Subject) is decomposed into that in Range (Object)
possible_way	*function*	*way*	*Function* can be achieved by the *way*
method_ function	*way*	*function*	*Way* contains *function* as sub(part)-functions to achieve the goal (whole) function

Document (adapted from http://www.fine-yasunaga.co.jp/english/home/wiresaw/index.htm)

What is Wire Saw?.....A wire (a piano wire of ϕ0.08 to 0.16mm) is wound around several hundred times along the groove of guide roller. Free abrasive grains (a mixture of grains and cutting oils) are applied to the wire while it keeps running. The abrasive grains rolled on the wire work to enable cutting of a processing object into several hundred slices at one time. It is mostly used to cut electronic materials.



Functional metadata

```
<funnotation:device rdf:about="http://ex.org/ex1.html#wire-saw">
  <funnotation:has_function>
    <funnotation:split_entity rdf:about="http://ex.org/ex1.html#cut">
     <funnotation:selected_way rdf:about="http://ex.org/ex1.html#grains">
      <funnotation:fricitional_way/></funnotation:selected_way>
    </funnotation:split_entity></funnotation:has_function>
</funnotation:device>
```

Fig. 4. Examples of metadata for a document of a wire-saw

named B-Unintended (Unintended behavior layer) whose role is to represent phenomena/behavior unintended by designers rather than function intended by them.

Moreover, concerning the functional terms, the system will be able to allow users to use other vocabulary of generic functions such as Generally-Valid Functions [21] and the functional basis [5]. In order to realize interoperability between our ontology and such a terminology, the discussion on the concept of function in broader sense in Section 3 plays a crucial role. In fact, the functional basis [5] includes *interpretational functions* in terms of the descriptive categories of function in Section 3. Using such conceptual categories of functions, we are currently developing an ontological mapping between our functional concept ontology and the functional basis. It enables us to integrate functional annotations based on different ontologies.

5. Related Work

We defined a function as a role of a behavior. In the literature, similar concepts are discussed. Chandrasekaran and Josephson 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" [2]. In EPISTLE Framework, the concept of facility is defined as a functional thing, capability to perform a function and a service [28]. The layered structure of our ontologies is similar to the PhysSys ontology [27]. It, however, has no ontology for functions from the teleological viewpoint. Some generic function sets with *is-a* hierarchy have been proposed such as generally-valid functions [21], "degree of abstraction" of functions [6] and the functional basis [5]. We define rich generic functions with clear operational relationship with objective behaviors. Similarly to the way of function achievement, a feature of function decomposition can also be found as a "means" in [18]. We defined *is-a* relations between generic ways of function achievement, and investigated how to organize them.

A functional modeling framework for the Semantic Web has been proposed in [14]. It is based on the functional basis [5] and is represented in the description logic (DL) for repository reasoning tasks. Our ontological work aims at providing comprehensive prescriptive guidelines for knowledge modeling (annotation) rather than the reasoning task. For example, our ontology provides the concept of "way of function achievement" as a key concept for capturing the functional structures of artifacts. The framework in [14] provides not such a concept but a representation schema in DL.

The generic tasks and the generic methods (PSMs) for problem-solving task research (e.g.,[51]) are similar to our generic functions and generic ways of function achievement for engineering domain knowledge, respectively. 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. 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. On generic "process", extensive research has been done elsewhere such as work in [7,23].

TRIZ (TIPS) theory provides some patterns (or strategies) for inventions based on the contradiction between two physical quantities [24]. 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.

6. Concluding Remarks

Ontological consideration on functionality of artifact and its application in the Semantic Web have been discussed. The role of ontologies is to provide semantic constraints to capture the target world consistently and controlled vocabulary for representation. The ontologies have been applied to modelling manufacturing machines, engineering plants, engineering products and manufacturing processes. The models have taken into account changes in thermal energy, flow rate, and ingredients of fluids, 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. The modelling framework currently cannot cope with the human's mental process, body movements (so-called therblig in Industrial Engineering), business processes, or software processes.

Acknowledgements. The authors would like to thank Kouji Kozaki, Munehiko Sasajima, Eiichi Sunagawa and Shinya Tarumi 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.

References

1. Borst, P., Akkermans, H., Top, J.: Engineering Ontologies. Int. J. of Human-Computer Studies 46 (1997) 365-406
2. Chandrasekaran, B., Josephson, J. R.: Function in Device Representation. Engineering with Computers 16 (2000) 162-177
3. Euzenat, J.: Eight Questions about Semantic Web Annotations. IEEE Intelligent Systems March/April (2002) 55-62
4. Gero, J.S., Kannengiesser, U.: The Situated Function-Behaviour-Structure Framework. In Proc. of Artificial Intelligence in Design '02 (2002) 89-104
5. Hirtz, J., Stone, R. B., McAdams, D.A., Szykman, S., Wood, K.L.: A Functional Basis for Engineering Design: Reconciling and Evolving Previous Efforts. Research in Engineering Design 13 (2002) 65-82
6. Hubka, V., Eder, W.E.: Functions Revisited. In Proc. of the 13th International Conference on Engineering Design (ICED 01) (2001), CD-ROM
7. ISO TC184/SC4/JWG8, Process Specification Language, http://www.tc184-sc4.org/ SC4_Open/ SC4_Work_Products_Documents/PSL_(18629)/ (2003)
8. Keuneke, A.M.: A Device Representation: the Significance of Functional Knowledge. IEEE Expert 24 (1991) 22-25

9. Kitamura, Y., Sano, T., Namba, K., Mizoguchi, R.: A Functional Concept Ontology and Its Application to Automatic Identification of Functional Structures. Advanced Engineering Informatics 16 (2002) 145-163
10. Kitamura, Y, Mizoguchi, R.: Ontology-based Systematization of Functional Knowledge. Journal of Engineering Design 15(4) (2004) 327-351
11. Kitamura, Y., Kashiwase, M. Fuse, M., Mizoguchi, R.: Deployment of an Ontological Framework of Functional Design Knowledge. Advanced Engineering Informatics 18 (2004) 115-127
12. Kitamura, Y., Koji, Y., Mizoguchi, R.: An Ontological Model of Device Function and Its Deployment for Engineering Knowledge Sharing. In Proc. of the First Workshop FOMI 2005 - Formal Ontologies Meet Industry (2005) CD-ROM
13. Koji, Y., Kitamura, Y., Mizoguchi, R., Ontology-based Transformation from an Extended Functional Model to FMEA. In Proc. of the 15th Int. Conf. on Engineering Design (ICED 05) (2005) 264.81
14. Kopena, J. B., Regli, W. C.: Functional Modeling of Engineering Designs for the Semantic Web, IEEE Data Engineering Bulletin, IEEE Computer Society, 26(4) (2003) 55-62
15. Kozaki, K., Kitamura, Y., Ikeda, M., Mizoguchi, R., Hozo: An Environment for Building/Using Ontologies Based on a Fundamental Consideration of "Role" and "Relationship". In Proc. of the 13th Int. Conf. on Knowledge Engineering and Knowledge Management (EKAW 2002) (2002) 213-218
16. Lee, J. Design Rationale Systems: Understanding the Issues. IEEE Expert 12(3) (1997) 78-85
17. Lind, M.: Modeling Goals and Functions of Complex Industrial Plants. Applied Artificial Intelligence 8 (1994) 259-283
18. Malmqvist J. Improved Function-Means Trees by Inclusion of Design History Information. Journal of Engineering Design 8(2) (1997) 107-117
19. Masolo, C., Vieu, L., Bottazzi, E., Catenacci, C., Ferrario, R., Gengami, A., and Guarino, N.: Social Roles and their Descriptions. In Proc. of the 9th Int'l Conf. on the Principles of Knowledge Representation and Reasoning (KR2004) (2004) 267–277
20. Miles, L.D.: Techniques of Value Analysis and Engineering. McGraw-hill (1961)
21. Pahl, G., Beitz, W.: Engineering Design - a Systematic Approach. The Design Council (1988)
22. Schreiber, G., et al.: Knowledge Engineering and Management - The Common-KADS Methodology, The MIT Press, Cambridge, MA (2000)
23. Sowa, J. F.: Top-level Ontological Categories, Int. J. of Human-Computer Studies 43(5-6) (1995) 669–685
24. Sushkov, V.V., Mars, N.J.I., Wognum, P.M.: Introduction to TIPS: a Theory for Creative Design. Artificial Intelligence in Engineering 9(3) (1995) 177-189.
25. Umeda, Y., Ishii, M., Yoshioka, M., Shimomura, Y., Tomiyama, T.: Supporting conceptual design based on the function-behavior-state modeler. Artificial Intelligence for Engineering Design, Analysis and Manufacturing 10 (1996) 275-288
26. Umeda, Y., Tomiyama, T.: Functional Reasoning in Design. IEEE Expert (1997) 42-48
27. van der Vegte, W.F., Kitamura, Y., Koji, Y., Mizoguchi, R.: 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
28. West, M.: Some Industrial Experiences in the Development and Use of Ontologies. EKAW 2004 Workshop on Core Ontologies in Ontology Engineering (2004) 1-14
29. W3C: OWL Web Ontology Language Reference, http://www.w3.org/TR/owl-ref/ (2004)
30. W3C: Resource Description Framework (RDF): Concepts and Abstract Syntax, http://www.w3.org/TR/rdf-concepts/ (2004)