**Proceedings of IDETC/CIE 2006**
**ASME 2006 International Design Engineering Technical Conferences &**
**Computers and Information in Engineering Conference**
**September 10-13, 2006, Philadelphia, Pennsylvania, USA**

**DETC2006-99131**

# AN ONTOLOGY-BASED ANNOTATION FRAMEWORK FOR REPRESENTING THE FUNCTIONALITY OF ENGINEERING DEVICES

**Yoshinobu Kitamura** **Naoya Washio** **Yusuke Koji**
**Munehiko Sasajima** **Sunao Takafuji** **Riichiro Mizoguchi**

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

## ABSTRACT

This paper proposes a metadata schema relating to functionality based on Semantic Web technology for the management of the information content of engineering design documents. The schema enables us to annotate web-documents with RDF metadata, which represents devices as having specific functions. The metadata provide a clear and operational semantics for the functional terms in documents. The metadata schema is based on our own functional ontologies, which we believe to provide effective guidelines for consistent functional annotation. These ontologies have been developed over many years and deployed successfully in industry.

We then demonstrate a document search system using the metadata, which enables us to retrieve web-documents using the types of function and relationships defined in the schema rather than more superficial terms. Such function-oriented management of information is especially useful in the conceptual design phase. It allows one to find previous cases of the same function in earlier designs and to find related patents.

We go on to discuss the following two issues on the interoperability of the functional knowledge. The first is automatic transformation of the functional model in our modeling framework into a chart-style form commonly used for the FMEA (Failure Modes and Effects Analysis) activity using a mapping between our ontology and an FMEA ontology. The second is translation of the functional model between our functional vocabulary and other taxonomy of function, that is, Functional Basis proposed by Stone et al. Another mapping between our functional vocabulary and Functional Basis can be done via a richer, generic reference ontology of functions. This mapping would aim at clarifying ontological differences between functional taxonomies and enabling translation between them.

## KEYWORDS

Design knowledge management, functionality, semantic web, ontology, metadata, conceptual design, fault analysis

## 1. INTRODUCTION

The importance of knowledge sharing for the task of design has been widely recognized. Like geometry data on CAD/CAE systems, it is important that the information content of design documents in a natural language are shared in order to represent the designer's intension, or the so-called design rationale [24]. One of the key concepts for representing design rationale is the functionality of devices [3][24]. A representation of the functionality of a device (a functional model) describes the designer's intended goals for the device and thus represents part of the design rationale [3]. The goal of this communication is to describe a framework to manage the content of engineering design documents with the aid of which engineers can access web-documents by specifying a function. Such function-oriented management is especially useful in the conceptual design phase; it enables one to find previous design cases that serve the same function and to find related patents.

While there are several requirements for sharing design documents, we concentrate on the following three issues. The first is to give clear and operational semantics to functional terms in documents. The current typical document management relies heavily on keyword and/or full-text searches based on lexical terms. But the semantics of the terms is not clear. Many terms (verbs) are used for the same function (and vice versa). Moreover, there is a difficulty in searching for a function in a specific context of a functional structure. For example, if one gives "slice" as a functional keyword, he or she will get documents about both slicing machines and sliced materials.

Our second concern is to provide effective guidelines to capture functions consistently. Although much research has been conducted on functionality in engineering design (e.g., [3][13][33][36][40]), it is difficult for engineers to avoid capturing functions in an ad hoc way. For example, to code "to weld metals" as a manufacturing machine's function as Value

 

Engineering does [30] is not only to name a function but also implies a certain way to achieve the goal, say, "that the metals' parts become fused". This issue, that of distinguishing "what to achieve" from "how to achieve it", is not terminological but ontological.

The third issue is interoperability of functional knowledge with other kinds of knowledge. We are especially concerned with knowledge about faults because it is closely related to functional knowledge. For example, in the FMEA (Fault Mode and Effects Analysis) activity [31], possible faults of each function and their effects are investigated. This enables a designer to add supplementary functions to prevent these faults. While it is difficult to retrieve them in an interoperable manner, knowledge about functionality and about faults are usually separately described. Moreover, there are some taxonomies of functions, such as Reconciled Functional Basis [12], which require interoperability with other taxonomies.

We adopt the Semantic Web technologies developed under a W3C's initiative[1] as technical standards to deal with the first issue, that is, to provide semantics of functional terms in documents. We recommend adding semantic annotation data representing functionality to web documents [41]. Functional metadata annotated to a web document describing an artifact shows the function of the artifact and/or those of its components in a machine-operational manner. We adopt RDF (Resource Description Framework) as a representation framework of metadata and OWL (Web Ontology Language) as a language for metadata schema which defines vocabulary for metadata as standards for Semantic Web.

With these standards, our main concern is to develop a functional metadata system that satisfies the second issue, that is, to provide effective guidelines for capturing functions consistently. We use our functional ontologies in our functional modeling framework to achieve this [16][17]. The framework consists of the categorization of types of functions and layered ontologies for representing these types. It provides the knowledge authors with a controlled vocabulary and guidelines for consistent and reusable knowledge and has been deployed successfully [18].

In this article, we propose a framework for annotating the functionality of engineering devices for sharing the content of technical documents. The framework is called *Funnotation* (abbreviation of FUNctional anNOTATION) hereafter. First, we propose a metadata schema based on our functional ontologies [16][17]. This includes the different ways of realizing or executing a function (the way of achieving a certain goal) and a hierarchy of generic functions with operational definitions. Then, we demonstrate a document search system using functional metadata. This allows engineers to search for design web documents by what they want to realize, i.e., by function.

We then discuss interoperability in the framework. As a generic mechanism for interoperability, the framework includes ontology-mappings, which shows the correspondences between terms in different ontologies and bridges the gap between them. Here, we discuss interoperability of our framework with FMEA [31] and with Reconciled Functional Basis [12]. We demonstrate a knowledge transformation system which generates chart-style documents of a type commonly used for FMEA activity from our models using an ontology mapping knowledge between our ontology and that of the FMEA. Another mapping between our functional vocabulary and Functional Basis can be done via a generic reference ontology of the types of functions. The resultant mappings would aim at clarifying the ontological difference between the functional taxonomies and at enabling translation between them. Part of the reference ontology has been discussed in [19][20]. This paper discusses how to use the reference ontology for interoperability among functional taxonomies and the mapping under investigation.

Section 2 shows an overview of our functional ontologies which have been reported previously in [16][17]. Section 3 discusses a metadata schema. Section 4 demonstrates software systems based on the *Funnotation* framework in the form of a document search system. Section 5 discusses interoperability. Related work is then discussed followed by some concluding remarks. The basic idea of functional annotation is discussed in [20]. This paper reports detail of the schema, the search system, and interoperability.

## 2. AN ONTOLOGICAL FRAMEWORK FOR SHARING KNOWLEDGE ABOUT FUNCTIONS

### 2.1. Functional Ontologies and Models

Our framework for representing functions [16][17] is shown in Fig. 1 as layers of ontologies, knowledge, and instance models. Basically, knowledge or a model in a certain layer is described in terms of the more general (and/or fundamental) types in the upper layer.

We define a function performed by a device as "a role[2] played by behavior of the device to achieve a specific goal under a *context of use*, based on a certain *capability* inherent to the device"[3], where behavior is objective temporal changes of physical quantity. The *context of use* can be basically determined by users or its higher-level components. The *capability of a device* for functioning is a property of an entity which represents what the entity can perform as function (*potential function*) when an appropriate context and appropriate input are given to the device. The potential functions according the capability are categorized into *essential function* and *accidental function*. The former is intended by a designer, which provides artifact's identity and thus names of many artifacts are derived from their essential functions. The latter is intended not by a designer but by a user in a particular context of use.

Such a definition shares characteristics of function such as intention-relatedness and context-dependence in the literature such as "means and ends" [26], F-B relationship [40] or "aims-means" [13]. In [15], function is defined as "the disposition of a certain entity reliably to act in such a way as to achieve a goal". This definition shares "goal-oriented"-ness of function and the device's inherent property of functioning with our definition above. This definition, however, excludes accidental functions. We emphasize multiple relationships between behavior (and its performer) and function by explicating contexts of use[4]. More-

[1] http://www.w3.org/2001/sw/

[2] Role is a technical term in Ontological Engineering. By role, we mean here such a thing that another thing plays (acts as) in a specific context and cannot be defined without mentioning external concepts [37]. 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) [29]. See [19] for discussion on characteristics of function as role.

[3] This definition of function has been refined from our previous ones in [16][19]. The extensive discussion with Prof. Barry Smith greatly helps the authors clarify our definition.

[4] This characteristic of artifacts (i.e., a device can perform multiple functions and a function can be performed by devices) is much different from biological organs, each of which has its unique function.

 

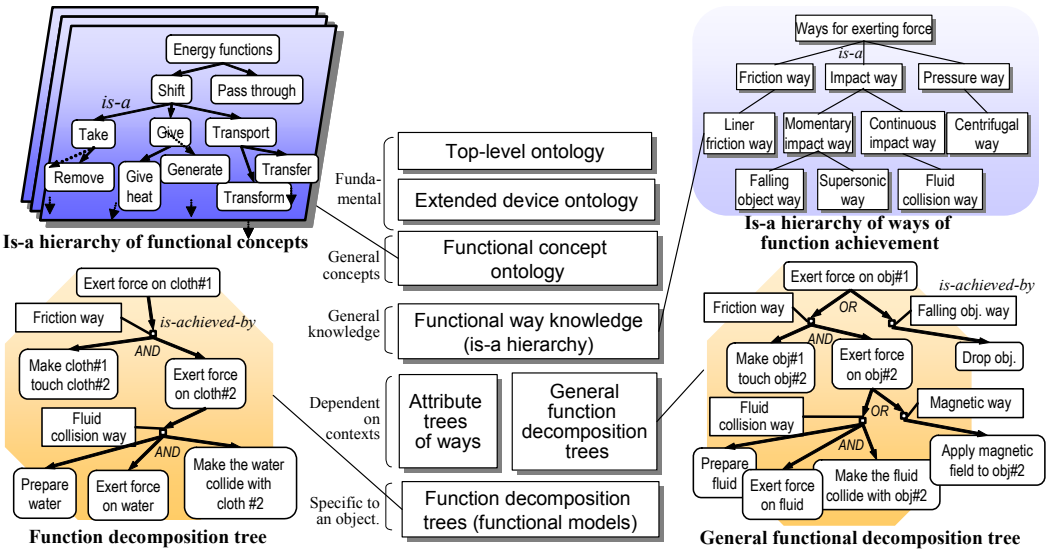


**Figure 1:** A layered framework of ontologies, knowledge, and models of functions.

over, we ontologically define relationships of function (i.e., *is-a* and *part-of* relations) and related concepts. In OGO [34], extensive discussion on primitive relations in biology is done. Here, we concentrate on relationships specialized to functionality of engineering devices.

At the bottom in Fig. 1, a function decomposition tree is a functional model of a specific device (in the figure, a washing machine). It represents that a required function (called a macro-function or a goal-function) can be achieved by specific finer-grained functions (called a micro-function or a method function). This is similar to the function decomposition in the German-style systematic design methodology [33], whole-part relation [26], and "degree of complexity" [13].

Our model includes types of background knowledge of functional decomposition such as physical principles and theories. These types are called "ways of function achievement". The way of achievement helps us detach "how to achieve" (way) from "what is intended to achieve" (function). For example, "to weld something" mentioned in the introduction should be decomposed into the "unifying function" and "fusion way". This increases the generality and capability of a functional model in that it accepts a wide range of ways to perform a function, such using a bolt and nut to achieve the same goal. Such alternatives are described in a general function decomposition tree shown in at the lower right of Fig. 1. This can be used when designers explore and investigate possible ways to achieve a specific required function.

We have developed an ontology of component functions (called a functional concept ontology, located at the third layer from the top in Fig. 1) [16] which is detached from ways of function achievement. It defines about 220 generic types for representation of functions in four *is-a* (a-kind-of) hierarchies. Pahl and Beitz defined highly-abstracted functions (called generally-valid functions) [33]. Hubka and Eder identify the hierarchy for the "degree of abstraction" of functions [13]. The hierarchy, however, includes terms such as "transportation by sea" [13], which imply a specific method of achieving a function in the same way as "welding". The recent efforts toward a standard taxonomy for engineering functions by the NIST Design Repository Project [12] are well established; however, they lack an operational relationship with behaviors and ontological specifications. In the functional concept ontology, each generic functional type has a clear operational relationship with objective behavior of a device. In order to capture functions consistently, it is based on an extended device ontology. Using these functional types as vocabulary, the function decomposition trees at the bottom of Fig. 1 are described.

The modeling of "way of function achievement" also helps us generalize concrete ways into generic ways and organize generic ways in *is-a* relations according to their principles (called functional way knowledge). Although the feature of function decomposition is also captured as "means" in [2][28], those works focus mainly on the function decomposition tree of a specific product. We focus on systematization (categorization) of general knowledge. Similar generic knowledge has been discussed in [9][40]. In our approach, based on a limited set of functional types, designers can explore explicit *is-a* hierarchies of functional ways knowledge.

A similar hierarchy of ontologies in the engineering domain is proposed in [1]. However, it does not include an ontology of functionality, which is our main concern.

## 2.2. Use and Deployment in Industry

Our framework contributes to making it easier to author consistent and reusable functional knowledge of a device. A functional model of a device can be a representation of design rationale. Functional knowledge can be used to redesign artifacts by changing the way of achieving their functions from that used in the original design.

Our framework has been deployed since May, 2001 into the plant and production systems engineering division of Sumitomo Electric Industries, Ltd. (hereinafter referred to as SEI) [18]. A knowledge management software named SOFAST has been developed based on part of the methodology and deployed since December, 2002. The targets of SOFAST are manufacturing equipment mainly used in semiconductor manufacturing processes including the wire-saw, a wafer polisher, and inspection machines. SOFAST has been used by 13 other companies since April, 2003. We summarize some of the usages and effects in deployment below.

One of the uses of the function decomposition tree is to clarify functional knowledge, which is implicitly possessed by

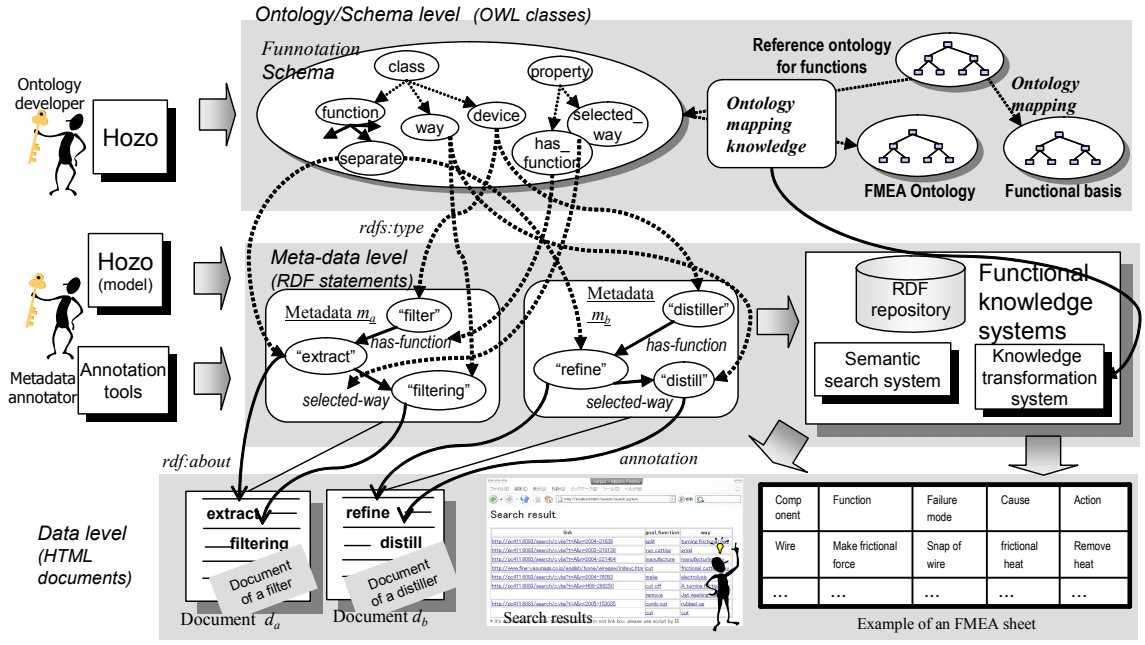


**Figure 2:** Overview of the *Funnotation* framework and systems.

each engineer, and share it with other engineers. The experiential evaluation by Sumitomo's engineers was unanimously positive. 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. This is because a function decomposition tree shows the designer's intentions on how to achieve the goal function and justifies design decisions, which are not included in the structural or behavioral models.

Explicit description of intentions is useful especially in the design review activity, where a team of designers double-check the original design and explore possible alternatives. The function decomposition tree shows alternative *ways* of achieving functions exhaustively for each (sub) function, their features in comparison, and reasons for adopting a specific way, or not, in one figure. The number of times the design reviews had to be done was reduced to one third after adopting our framework.

Such a deep understanding contributes to redesigning and solving problems concerning equipment. For example, after four months of investigation an engineer was not able to reduce the time a machine requires to polish semiconductor wafers by adjusting the known working parameters. By referring to that of other machine (a wire saw to slice ingots), he was able to describe its function decomposition tree. Although these two devices have different main functions, he found the shared function "to maintain a large friction coefficient". By comparing its sub-functions, he became aware of an implicit function; "to place diamond powder between wafers and the table" of a guide ring and its parameters to obtain a high friction coefficient. Eventually, he reduced the necessary operation time to 76% within three weeks, which was better than the initial goal.

This example shows an effect of reusing functional knowledge between different devices in redesign. It provides the engineer with stimuli for reflection by reusing knowledge of other equipment. The ontologies help engineers describe reusable knowledge by providing fundamental types such as ways of function achievement and generic functions as shared vocabulary.

## 3. ANNOTATION ABOUT FUNCTION

### 3.1. Overview of the *Funnotation* Framework

In the semantic web context, our ontology can be used as a metadata schema for engineering documents as shown in Fig. 2. The metadata schema based on the ontology for functional annotation is called the *Funnotation* schema. The ontologies in our framework provide hierarchies of classes (types) as metadata schema in OWL. The framework enables us to describe metadata representing functionality of engineering devices appearing in the target engineering documents. The metadata about functionality as RDF statements in XML syntax are described as instances of those classes. The functional way knowledge provides hierarchies of classes for representing patterns of "how to achieve a function" of a device.

For example, two metadata $m_a$ and $m_b$ shown in the center of Fig. 2 are annotated to the two technical documents $d_a$, and $d_b$ respectively, shown in the bottom part of the figure. A part of $m_a$ shows that the device appearing in the document $d_a$ (a filter) has a *separating* function. It is represented as an instance of the "separate" function-class defined in the *Funnotation* schema. This metadata is annotated to the term "extract" in $d_a$. The metadata $m_b$ shows that the distiller (the device mentioned in the document $d_b$) has the same *separating* function. It is, however, annotated to the term "refine" in $d_b$. In this manner, functional metadata show a device's functions independently of the terms in documents and indicates pointers (URLs) to the original documents and/or terms. In addition to functions, the metadata include the ways of function achievement, i.e., the filtering way in $m_a$ annotated to $d_a$ and the distilling way in $m_b$ annotated to $d_b$. In this manner, the function metadata show how to achieve a function, i.e., in this case, two different ways to achieve the same function. Detail of such metadata is discussed in the following sections.

By querying such functional metadata, a semantic search system is designed to provide access to the annotated documents based on classes of functions and/or the relationship of functions. Using the example in Fig. 2, if an engineer specifies

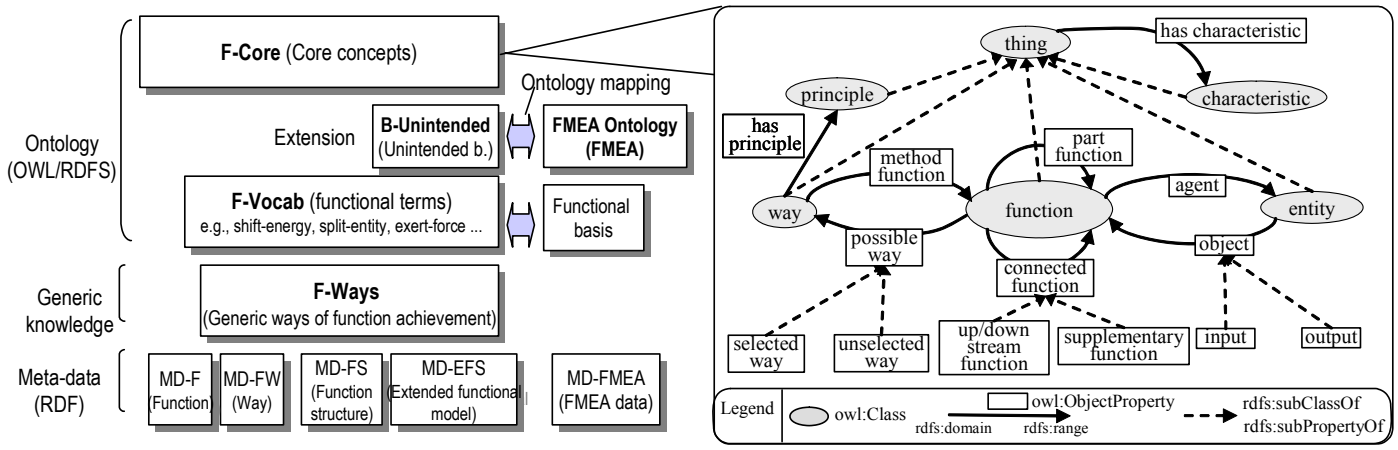


**Figure 3:** Layers of *Funnotation* Schema and the classes and the properties of F-Core (portion).

the "separate" function as a goal function in a query, the system provides hyperlinks to the both documents $d_a$ and $d_b$. The details of searches on functional metadata are discussed in Section 4.2.

The *Funnotation* framework realizes interoperability with other kinds of knowledge based on *ontology mapping*. There are two ways of mappings, i.e., *direct mapping* and *reference mapping*. The former shows direct correspondences between types in two ontologies. Section 5.1 discusses a direct mapping between our functional ontology and the FMEA ontology and automatic FMEA generation (transformation) from an integrated model of function and fault.

By reference mapping, we mean relationships via a richer, generic reference ontology of functions, which includes various types of function in different definitions [19]. As an example, we are currently investigating ontology mapping to Reconciled Functional Basis [12]. Section 5.2 shows an overview of the reference mapping.

From the general viewpoint of metadata research, using the terms of categorization in [8], the functional metadata can be regarded as "content descriptors" like keywords or "logical structure" of "content representation" like a summary or an abstract. By logical structure, we mean the relationship among functions such as functional decomposition. Functional metadata explicates the design rationale underlying design documents such as design drawings.

### 3.2. *Funnotation* Schema

The proposed metadata schema, called *Funnotation* Schema, has been built with the intention of annotating web resources about artifacts from their functional aspects. The schema consists of layers (sub-schemata) such as F-Core, B-Unintended, F-Vocab and F-Ways schemata as shown in Fig. 3. In F-Core schema, core types such as *entity, function* and *way* are defined together with properties among them. Figure 3 shows a part of classes and properties in F-Core. An *entity* represents a thing that exists in the physical world. It includes *device, stuff* (e.g., material, liquid, gas) and *energy* as its subtypes.

A *function* is performed by an entity (called agent) and changes another entity(s) (called object). The *agent* property is a relation between an (subclass of) entity and a function, where the entity can perform the function as an agent. That the value of a physical quantity of an *entity* is changed by a *function* is represented by the *object* property. The *input* and *output* property are sub-types of the *object* property.

A function can be achieved by finer-grained functions, which are represented by *part_function* properties. They form a function decomposition tree as discussed in Section 2. The principle of the achievement is conceptualized as an instance of the *principle* class. It is associated with *way*, which represents a way of function achievement discussed in Section 2. A way is associated with a *function* to be achieved (using *goal_function* property), *functions* achieving the goal function, (*method_function* properties) and its *principle* (*has_principle* property). This implies that the *goal_function* can be achieved by the series of the *method_functions* in the *way*, which is represented as *possible_way* property (the inverse property of *goal_function* property). The *selected_way* and *unselected_way* are subtypes of the *possible_way* property according to the result of conceptual design for a specific device. The series of method_functions in a specific *way* can be described using *connected_function* or its subtypes; *up/down_stream_function* properties. A *supplimentary_function* is a kind of a method function which contributes to the improvement of quality or efficiency of another function.

Verbs such as *convey* and *separate* are defined in F-Vocab schema as subclasses of *function*. Those terms come from the functional concept ontology. In F-Ways schema, generic function-achievement ways such as *frictional_way* for exerting force are defined as a subclass of *way* class. The definition of each way of function-achievement is composed of the principle on which the achievement is based, the goal function (achieved function) and sub-functions which collectively constitute the way.

### 3.3. *Funnotation* Metadata

The *Funnotation* schema enables users to describe functional metadata (called *Funnotation* metadata) with RDF 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. While (3) and (4) correspond to a full model of the functional structure, (1) and (2) correspond to "indexing" information (content descriptors) representing some portion of the full model.

Figure 4 shows functional metadata added to the document about a wire saw, which is a manufacturing machine to slice

semiconductor ingots by friction using a moving wire. In Fig. 4, the saw's function is described as "split" as an instance of the splitting function class *(Funnotation:split)*. It is annotated to the term "cut" in the document. The agent (performer) of the function, i.e., the wire-saw, is described using the *agent* property.

In addition to metadata about functions, one can describe ways of function achievement used in the device. Figure 4 shows that an instance of *funnotation:frictional_way* is linked to the *splitting function* instance via the *selected_way* property to demonstrate that the wire saw achieves its main function using frictional force. Thanks to our functional ontologies, an author can describe functions and ways separately. The metadata in Fig. 4 includes a description about a *method_function* and a *supplimentary_function* as well, which represents "make contact abrasive grains to wire" as a supplementary function (for high friction coefficient).

In total, the metadata in Fig. 4 shows a (part of) function decomposition tree of the wire-saw. It is associated to the terms in the document but is independent of the lexical expressions and machine-operational with a clear relationship. Note that the richness of the metadata in Fig. 4 is not necessary. An author can describe simpler metadata according to his/her needs.

Furthermore, the general function decomposition tree can be regarded as another kind of metadata. It can consist of several ways of function achievement in different devices. Thus, it can give a combined summary of some documents from the viewpoint of functionality.

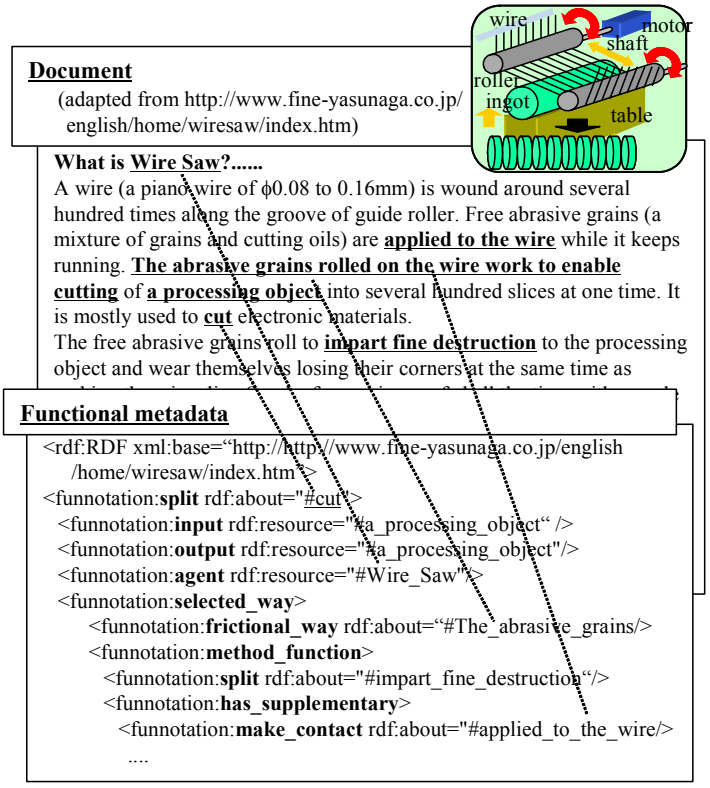


**Figure 4:** An example of metadata for a document of a wire-saw (portion).

## 4. *FUNNTOTATION* SYSTEM

### 4.1. Ontologies and Metadata Annotation

We implemented our ontological framework using our ontology development environment named Hozo [37]. Hozo can export ontologies and instance models in OWL, which can be used as a metadata schema. The extended device ontology, the functional concept ontology, and the functional way knowledge are exported as classes in OWL. A specific way knowledge is represented as a sub-class of the "way" class in hierarchies by restricting (specializing) its range to a specific function class in the functional concept ontology.

Much research has been conducted on annotating web-documents with metadata elsewhere. The annotation authoring system itself has been developed elsewhere [41] and is not one of our main concerns. Currently, we use two tools for functional annotation: one is to describe an instance model in Hozo and export it as a RDF file. The other is to use OntoMat-Annotizer[5] with the schema in OWL exported by Hozo. Functional metadata in RDF is then obtained.

### 4.2. Semantic Search for Functional Metadata

As shown in Fig. 2, the Funnotation Semantic Search System interprets written metadata, searches for relevant documents and displays hyperlinks for documents. With a web browser, engineers input search conditions such as goal-function (function that they want to achieve), input objects (objects that a function is for), and more. The system selects metadata conforming to the search condition from all metadata that have been collected by a crawling engine and stored in a metadata repository, and displays hyperlinks indicated by the selected metadata.

This system consists of a user interface on a web browser and a server module on a web server. The former is a user-friendly interface specialized to input search conditions to retrieve metadata about functional knowledge. It is implemented using HTML, CSS, and JavaScript. The latter is a module which retrieves search results after sending a query to the metadata repository and transforms the results into a HTML document. It is implemented by Java and uses Tomcat with a HTTP server, Jena[6] to operate the RDF repository, and SPARQL[7] as a RDF query language.

As shown in Fig. 5, the users input the search condition using a graphical pattern which is at the upper part of that interface. Here, in the same manner of the functional decomposition tree, they set a goal-function, method-functions (functions to achieve the goal-function), input and/or output object, and more, as search conditions. At the bottom left of the interface, there are "Search Situation Help" buttons to specify search conditions according to the users' situations. At bottom right, there are "Search output filter" check-boxes for setting post search conditions in addition to the condition in the graphical pattern.

In this system, users can make various patterns of queries. The most typical is where a user specifies a function class as a goal-function, and then gets documents which describe devices achieving the function. For example, let us consider a user who would like to know the ways to separate a semiconductor ingot. First, the user clicks "I'm looking for ways to achieve a goal function" in the search situation help area. Next, he/she selects the "separate" class from the list of the functional vocabulary defined in F-Vocab. The "separate" class represents that an entity is divided into more than two sections. As a result of

[5] http://annotation.semanticweb.org/ontomat/index.html

[6] Jena, A Semantic Web Framework for Java", http://jena.sourceforge.net/

[7] SPARQL Query Language for RDF, http://www.w3.org/ TR/rdf-sparql-query

these search conditions shown in Fig. 5, he/she gets the search results shown in Fig. 6(1). In Fig. 6(1), the leftmost column shows hyperlinks for documents. The center column indicated as "goal-function" shows the words in documents which are annotated as the "separate" function class and as a subject of "selected_way" (or "possible_way") property. The rightmost column shows the terms annotated as a way.

This example shows that users can search for documents with a generic type of function independently of the original words in the documents. For example, in document (a) in Fig. 6(1) about a manufacturing machine to cut semiconductor panels, the term "split" is used as its function. Document (b) describes a reactor as something to "make" a chemical object by electrolysis. Document (c) describes a slicing machine to "cut off" wafers from an ingot. It includes a component's function to "remove" scrapings as well. All such functions were regarded as "separate" according to the definition in the functional concept ontology.

Even if "separate" is written in a document, however, the document is not retrieved for the same query mentioned above when it is not annotated as a goal-function of a way. For example, a document about a device which assembles "separated" semiconductors does not match the search condition, because the term "separate" is not annotated as a goal-function.

Moreover, thanks to a hierarchical structure of the types of generic functions in *is-a* relation, users can change the level of abstraction of the search condition. In the example described above, the user retrieved the search results by specifying "separate" as the goal-function. The search results include not only "separate" but also its subclasses such as "split", "take_out" and "decompose". If the user specifies "take_out" as a goal function, he/she will get the result shown in Fig. 6(2). It includes documents (b) and (c), because "make" in (b) and "remove" in (c) are categorized into the 'take_out' function for getting a specific part (or ingredient) from a major part, while "split" in (a) is categorized into the 'split' function for getting small pieces of the same kind of object.

The result shown in Fig. 6(2) includes documents about components having the taking-out function. If the user wants to exclude them, he/she can specify it using "exclude the goal function that is also a method function of other goal function" in the "Search output filter". Then, the result in Fig. 6(3) is obtained. It includes only document (b) which describes a taking-out function ("make") as the whole function, while it excludes document (c) which describes a taking_out function ("remove") of a component.

Users can also search for supplementary functions. As introduced in Section 3.2, a supplementary function is a method function that contributes to prevention of faults, improvement

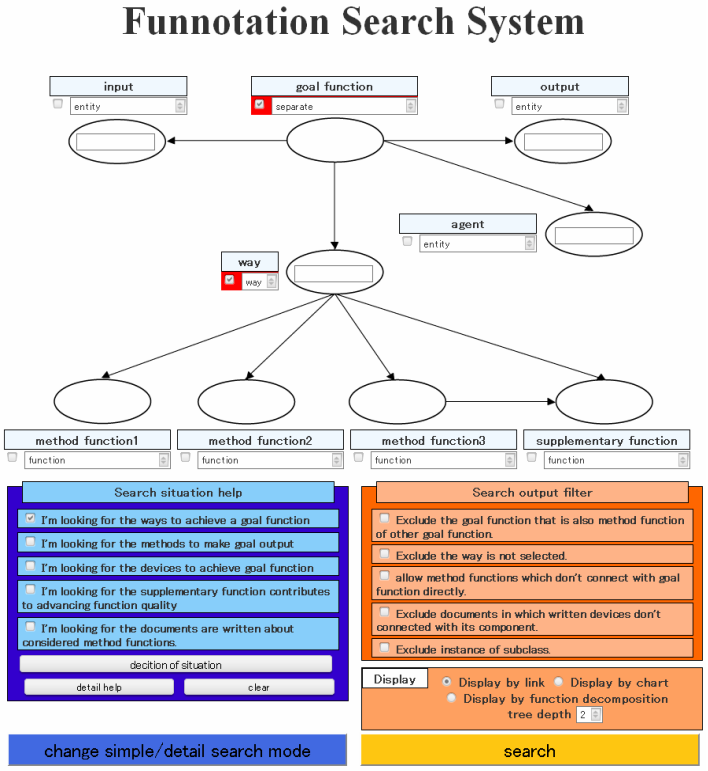


**Figure 5:** The interface of *Funnotation search system.*

(1)

| | link | goal_function | way |
|---|---|---|---|
| (a) | http://pc411:8083/search/c.vbs?t=A&n=2004-31639 | split | rotatiing frictional way |
| | http://pc411:8083/search/c.vbs?t=A&n=2003-318138 | cutting | grind |
| | http://www.fine-yasunaga.co.jp/english/home/wiresaw/indexc.htm | cut | frictional cutting way |
| | http://pc411:8083/search/c.vbs?t=A&n=2004-221464 | manufacture | manufacturing way |
| (b) | http://pc411:8083/search/c.vbs?t=A&n=2004-76053 | make | electrolysis |
| (c) | http://pc411:8083/search/c.vbs?t=A&n=H08-298250 | cut off | A rotatiing friction way |
| | | remove | Jet washing way |
| | http://pc411:8083/search/c.vbs?t=A&n=2005-153035 | comb out | rubbed up |
| | | cut | cut |

(2)

| | link | goal_function | way |
|---|---|---|---|
| (b) | http://pc411:8083/search/c.vbs?t=A&n=2004-76053 | make | electrolysis |
| (c) | http://pc411:8083/search/c.vbs?t=A&n=H08-298250 | remove | Jet washing wa |
| | http://pc411:8083/search/c.vbs?t=A&n=2005-153035 | comb out | rubbed up |

(3)

| | link | goal_function | way |
|---|---|---|---|
| (b) | http://pc411:8083/search/c.vbs?t=A&n=2004-76053 | make | electrolysis |

**Figure 6:** Examples of search results; (1)-(3).

(4)

| | link | goal_function | way | supplementary function |
|---|---|---|---|---|
| (a) | http://pc411:8083/search/c.vbs?t=A&n=2004-31639 | split | rotatiing frictional way | harden |
| | http://www.fine-yasunaga.co.jp/english/home/wiresaw/indexc.htm | cut | frictional cutting way | applied to the wire |
| | http://pc411:8083/search/c.vbs?t=A&n=2004-221464 | manufacture | manufacturing way | was aligned |
| (b) | http://pc411:8083/search/c.vbs?t=A&n=2004-76053 | make | electrolysis | are along with |
| (c) | http://pc411:8083/search/c.vbs?t=A&n=H08-298250 | cut off | A rotatiing friction way | remove |

(5)

| | link | goal_function | way | supplementary function |
|---|---|---|---|---|
| (c) | http://pc411:8083/search/c.vbs?t=A&n=H08-298250 | cut off | A rotatiing friction way | remove |

**Figure 7:** Examples of search results; (4) and (5).

of quality or of efficiency of another method function. Let us suppose a situation in which a designer attempts to redesign a slicing machine to solve a problem caused by scrapings. Three possible (imaginary) cases of search query according to the designer's preference are as follows. First, suppose the designer searches for documents for a drastic change of the current design. The search query shown in Fig. 5 and its result shown in Fig. 6(1) might help him/her, because the result might include possible alternative ways of function achievement without scrapings such as cutting ceramics using lateral pressure.

As a second option, suppose the designer would like to keep as much of the current design as possible and is looking for the ways to avoid difficulties in the separating function as a method function by adding a supplementary function. He/she might check the "supplementary function" box (with "function" class for any functions) and set "separate" function as a "method function" to be improved by the supplementary function in addition to the condition shown in Fig. 5. Figure 7(4) shows a result for this query which includes some supplementary functions to improve the separating function. For example, document (a) explains a way that hardens the target objects with ultraviolet rays before slicing to avoid anomaly caused by the scrapings. In document (b), a supplementary function is to align the separated object.

As the third case, suppose the designer is about to decide a specific way for the re-design, say, to "take_out" the scrapings so he/she sets "take_out" as a supplementary function and then gets the result shown in Fig. 7(5). The results contain only document (c) in which scrapings are taken out by flowing-fluid as a supplementary function.

### 4.3. Usage for Design

The *Funnotation* search system helps a designer access annotated documents from the viewpoint of function. It can be used mainly in the conceptual design phase. A designer can explore possible alternative ways for achieving a function by specifying the function as a goal function. He or she can find previous design cases that serve the same function and then might get stimuli for alternative design. Related patents also can be retrieved from a required function, input/output objects, or the way of function achievement. In embodiment design phase, by specifying a function, a designer can retrieve a document about a component which can perform the function.

For such usages, it is crucial that users can distinguish "function" from "way of function achievement". For example, for the documents and metadata shown in Fig. 2, when some user looks for the method to achieve the goal which is taking out something from another thing, he or she can discover the documents about not only "the filter way" but also "the distillation way". Moreover, as discussed in the previous section, this system makes it possible for designers to retrieve documents according to their situation and preference by referring to ways and supplementary functions as search conditions.

## 5. ONTOLOGY MAPPING FOR INTEROPERABILITY

The core of the *Funnotation* framework discussed thus far is based on our functional ontologies discussed in Section 2. The framework provides interoperability with other kinds of knowledge based on ontology mapping. Ontology mappings to FMEA and to Functional Basis are discussed in Section 5.1 and 5.2, respectively.

### 5.1. Extension to Fault Knowledge and Knowledge Transformation to FMEA

Knowledge about fault and anomaly is closely related to functional knowledge. For example, artifacts include a function that prevents possible faults. For explicating the design rationale of such functions, we need integrated models of both functions and anomaly. We developed an integrated model as an extension of the functional ontologies discussed in Section 2 [21]. The upper part of Fig. 8 shows an integrated model of a wire-saw. It shows a design rationale of its cooling function (by putting fluid over the wire), that is, to remove heat caused by the friction of the wire, which is a possible cause of the wire breaking.

The typical chart-style forms (documents) used for FMEA include a part of the same information such as possible faults of functions and their effects. For interoperability of our framework with FMEA, we have developed a transformation system to generate FMEA documents from an integrated model [22]. The system is based on an ontology mapping between the extended functional ontology and an FMEA ontology. An FMEA ontology consists of types which correspond to terms such as "potential failure modes" and "potential effect(s) of failure mode" in the chart-style form of FMEA. It shows the correspondences between types in our functional ontology and those of the FMEA ontology. This mapping required an ontology alignment because of the unstable use of types in FMEA. For example, "failure mode" is defined as "the manner by which a failure is observed" [9]. In FMEA documents in practice, however, "inefficient", "vibration" and "fracture" are found. These correspond to *function defect*, *trigger phenomenon* of faults, and *direct causative phenomenon of function defect* in our ontology, respectively [22]. As a result of ontology alignment, we describe a "corresponding-to" relation between "failure mode" in FMEA and *function defect* in our ontology.

We have developed a module for model transformation embedded in Hozo, which is a general mechanism for model

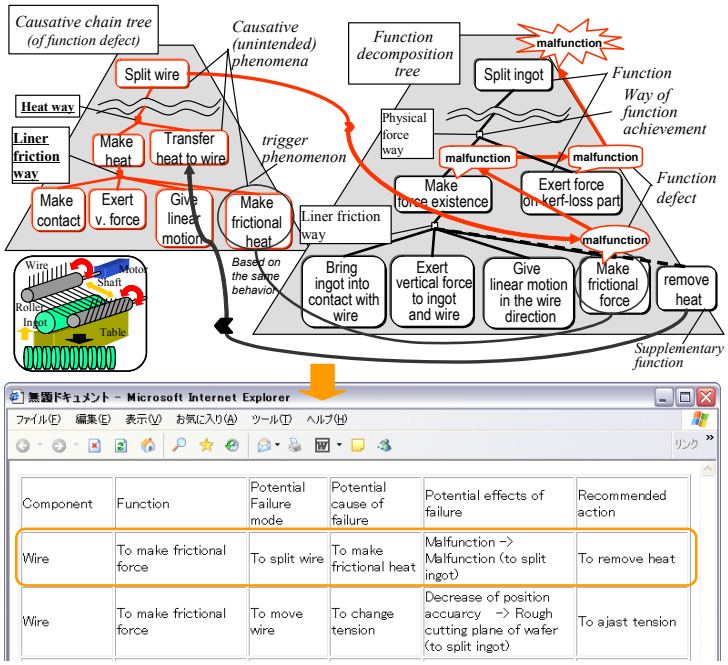


**Figure 8:** Example of knowledge transformation from an extended functional model to a FMEA document.

transformation using ontology mappings [22]. Given the ontology mapping between our ontology and FMEA, FMEA documents are generated from an integrated model as shown Fig. 8.

In the *Funnotation* framework, an additional schema called the B-Unintended (Unintended behavior layer) Schema is used. The F-Core schema is generalized as well. Using this schema, authors can describe metadata representing causative process of faults, faults, their effects, and how a supplementary function prevents a fault. Engineers can access metadata in the conventional form (i.e., FMEA in this case) their task (i.e., reliability analysis) requires.

### 5.2. Mapping with Functional Basis

Reconciled Functional Basis [12] is a well-developed taxonomy of functions. As a core of the *Funnotation* framework, the F-Vocab schema provides a functional vocabulary. It is important to realize interoperability between the two functional taxonomies.

The mapping between our functional vocabulary and Functional Basis can be done via a generic reference ontology of the types of functions, as shown in Fig. 2. In our functional ontology discussed in Section 2, the type of function is strictly defined from the device-centered viewpoint, which is intended to prescribe guidelines to functional annotation. Other types of function, however, are used elsewhere. The reference ontology of functions defines types of functions in a broader sense, which includes the types of functions other than the device-oriented function [19]. Each term (function) in both taxonomies is classified into a type of function in the reference ontology. Many functions in both taxonomies are classified into a type of function named "flowing-object function", which represents a device as a black-box that changes the state (a value of physical quantity at a time point) of objects (or stuff) flowing through that device. Such functions categorized into the same type can be associated with each other in the same manner mentioned in Section 5.1. Even for such functions, there are many mismatches due to difference of categorization in addition to lexical difference. The concrete mappings are currently under investigation.

Some functions, however, are categorized into different types. For example, Functional Basis includes an *interpretational function* [19] which requires human's cognitive interpretation. The reference ontology for functions would aim at clarifying such ontological differences between the functional taxonomies and at enabling translation between them.

## 6. RELATED WORK

In Section 2, we discussed our definition of function and some similar definitions in the literature. Here, we discuss other remaining definitions of function. In contrast to our device-oriented function, *environment function* as an effect on the surrounding world [4] and *purpose* [9][13][14][26] related to human-intended goals are investigated. The secondary function [33] and assisting function [13][14] are similar to our supplementary function discussed in Sections 3.1, 4.2 and 5.1. Garbacz defines participation functions according to categories of perdurants (i.e., process) in an ontological meta-theory [10].

A functional modeling framework for the Semantic Web has been proposed in [23]. It is based on Functional Basis [12] 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 introduces "way of function achievement" as a key type for distinguishing a function from its realization. The framework in [23] does not provide such a type but rather a representation schema in DL. Without such a type, the functional model in [23] is directly associated with components (i.e., carrier/agent of function) as a part of realization.

Automatic (or systematic) generation of FMEA documents from behavior or functional models such as FMAG [39] and Advanced FMEA [35] has been investigated. In addition to fault modes and effects in those models, our integrated model includes a detailed causative process of faults [21].

The ontology-based integration and interoperability among design knowledge have been investigated from early 1990's such as PACT [7] and KIEF [42]. They mainly focus on generic interoperable mechanism among agents and/or engineering tools. Product data exchange based on ontology has been proposed in [5]. An ontology of a specific product family in OWL DL also has been developed for product data management (PDM) [32]. An ontology-based design knowledge modeling is discussed in [27]. We aim at generic and richer ontology of function and clear conceptualization of related types. DAEDALUS knowledge engineering framework [25] is a framework for knowledge sharing between design and diagnostic tasks. It uses a simple top-level ontology and an ontology filter for model transformation. We have developed rich functional ontologies and use ontology mapping.

TRIZ (TIPS) theory provides some patterns (or strategies) for inventions based on the contradiction between two physical quantities [38]. We did not concentrate on design strategies but on modeling schema. TRIZ theory also concentrates on physical principles (effects), although we established a clear relationship between physical principles and functional structures.

## 7. CONCLUSIONS

This paper proposed a framework of semantic annotation about functionality for the management of the information content of engineering documents. A proposed metadata schema enables authors to annotate web-documents. Our functional ontology as a basis of the schema, we believe, provides a semantic constraint for annotating documents consistently. A document search system based on functional annotation has been developed. It helps engineers retrieve documents using functions to be achieved by devices in the documents and their relationship. The interoperability of our ontology with FMEA and with Functional Basis was also discussed.

## ACKNOWLEDGMENTS

The authors are most grateful to Prof. Barry Smith for many valuable comments and extensive discussion. The authors thank Dr. Kouji Kozaki and Mr. Masataka Takeuchi for their contribution.

## REFERENCES

[1] Borst, P. Akkermans, H., and Top, J, 1997, "Engineering Ontologies", Human-Computer Studies, 46(2/3), pp. 365-406.

[2] 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*.

 

[3] Chandrasekaran, B.; Goel, A. K.; Iwasaki, Y, 1993, "Functional representation as design rationale" *Computer*, **26**(1), pp. 48-56.

[4] Chandrasekaran, B, Josephson, J.R., 2000, "Function in Device Representation", *Engineering with Computers*, **16**(3/4), 162-177.

[5] Christel D. and Parisa C., 2002, "Product Data Exchange Using Ongologies", In *Proc. of Artificial Intelligence in Design* (AID2002), Cambridge, UK, pp. 617-637.

[6] Collins. J. A., 1993, "Failure of materials in mechanical design: analysis, prediction, prevention", Wiley Interscience.

[7] Cutkosky, M.R., et al., 1993, "PACT: An Experiment in Integrating Concurrent Engineering Systems", *Computer*, January:28-37.

[8] Euzenat, J., 2002, "Eight Questions about Semantic Web Annotations", *IEEE Intelligent Systems* March/April, pp. 55-62.

[9] Gero, J.S., Kannengiesser, U., 2002, "The Situated Function-Behaviour-Structure Framework", In *Proc. of Artificial Intelligence in Design '02*, pp. 89-104.

[10] Garbacz, P., 2005, "Towards a Standard Taxonomy of Artifact Functions", In *Proc. of the First Workshop FOMI 2005 - Formal Ontologies Meet Industry*, CD-ROM.

[11] Hawkins, P. G., Woollons, D. J., 1998, "Failure Modes and Effects Analysis of Complex Engineering Systems using Functional Models", *Artificial Intelligence in Engineering*, **12**, pp. 375-397.

[12] 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**, pp. 65-82.

[13] Hubka, V., Eder, W.E., 1988, *Theory of Technical Systems*, Berlin: Springer-Verlag.

[14] Hubka, V., Eder, W.E., 2001, "Functions Revisited", In *Proc. of ICED 01*.

[15] Johansson, I., Smith, B., Munn, K., Tsikolia, N., Elsner, K., Ernst, D., and Siebert, D., 2005, "Functional Anatomy: A Taxonomic Proposal", *Acta Biotheoretica*, 53(3), pp. 153-66

[16] Kitamura, Y., Sano, T., Namba, K., and Mizoguchi, R., 2002, "A Functional Concept Ontology and Its Application to Automatic Identification of Functional Structures". *Advanced Engineering Informatics* **16**(2), pp. 145-163.

[17] 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), pp. 164-166.

[18] 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.

[19] Kitamura, Y., Koji, Y., Mizoguchi, R., 2005, "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*, CD-ROM.

[20] Kitamura, Y., Washio, N., Koji, Y., Mizoguchi, R., 2006, "Towards Ontologies of Functionality and Semantic Annotation for Technical Knowledge Management", *New Frontiers in Artificial Intelligence: Proceeding of the 19th Annual Conferences of the Japanese Society for Artificial Intelligence*, LNAI 4012, to appear.

[21] Koji, Y., Kitamura, Y., Mizoguchi, R., 2004, "Towards Modeling Design Rationale of Supplementary Functions in Conceptual Design", In *Proc. of Tools and Methods of Competitive Engineering - TMCE 2004 (TMCE2004)*, pp. 117-130.

[22] Koji, Y., Kitamura, Y., Mizoguchi, R., 2005, "Ontology-based Transformation from an Extended Functional Model to FMEA", In *Proc. of ICED 05,* 264.81.

[23] Kopena, J. B., Regli, W. C., 2003, "Functional Modeling of Engineering Designs for the Semantic Web", *IEEE Data Engineering Bulletin*, IEEE Computer Society, **26**(4), pp. 55-62.

[24] Lee, J., 1997, "Design Rationale Systems: Understanding the Issues", *IEEE Expert* **12**(3), pp. 78-85.

[25] Lee, B. H., 2001, "Using FMEA models and ontologies to build diagnostic models", *Artificial Intelligence for Engineering Design, Analysis and Manufacturing*, **15**, pp. 281-293.

[26] Lind, M., 1994, "Modeling goals and functions of complex industrial plants", *Applied Artificial Intelligence* **8**, pp. 259-283.

[27] Li, Z., Anderson, D. C., Ramani, K., 2005, "Ontology-based Design Knowledge Modeling for Product Retrieval", In *Proc. of ICED 2005*, 462.1.

[28] Malmqvist, J., 1997, "Improved Function-means Trees by Inclusion of Design History Information", *Journal of Engineering Design* **8**(2), pp. 107-117.

[29] 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.

[30] Miles, L. D., 1961, Techniques of value analysis and engineering. McGraw-hill.

[31] Military Standard, 1980, Procedures for Performing a Failure Mode, Effects and Criticality Analysis, MIL-STD-1629A.

[32] Nanda, J., Thevenot, H. J., Simpson, T. W., Kumara, S. R. T., Shooter, S. B., 2004, "Exploring Semantic Web Technologies for Product Family Modeling", In *Proc. of ASME DETC 2004*, DETC2004-57683.

[33] Pahl, G., and Beitz, W., 1998, *Engineering Design - a Systematic Approach*. The Design Council.

[34] Smith, B., Ceusters, W., Klagges, B., Köhler, J., Kumar, A., Lomax, J., Mungall, C., Neuhaus, F., Rector, A. L., Rosse, C., 2005, "Relations in biomedical ontologies", *Genome Biology*, 6:R46.

[35] Steven, K., Peder, F., and Ishii, K., 1999, Advanced Failure Modes and Effects Analysis of Complex Processes, *Proc. of the ASME Design for Manufacturing Conference*.

[36] Stone, R. B., Chakrabarti, (eds.), 2005, Special Issues: Engineering applications of representations of function, *Journal of Artificial Intelligence for Engineering Design, Analysis and Manufacturing (AI EDAM)*, **19**(2 and 3).

[37] Sunagawa, E., Kozaki, K., Kitamura, Y., Mizoguchi, R., 2005, "A Framework for Organizing Role Concepts in Ontology Development Tool: Hozo", Papers from the AAAI Fall Symposium "Roles, an Interdisciplinary Perspective" AAAI TR FS-05-08, pp. 136-143.

[38] Sushkov, V.V., Mars, N.J.I., Wognum, P.M., 1995, "Introduction to TIPS: a Theory for Creative Design", *Artificial Intelligence in Engineering,* **9**(3), pp. 177-189.

[39] Teoh, P.C. and Case, K., 2004, "Modelling and Reasoning for Failure Modes and Effects Analysis Generation'", *Proceedings of the Institution of Mechanical Engineers, Part B: Journal of Engineering Manufacture*, **218**, pp 289-300.

[40] Umeda, Y., Ishii, M., Yoshioka, M., Shimomura, Y., and Tomiyama, T., 1996, "Supporting conceptual design based on the function-behavior-state modeler" *Artificial Intelligence for Engineering Design, Analysis and Manufacturing* **10**, pp. 275-288.

[41] Uren, V., Cimiano, P., Iria, J., Handschuh, S., Vargas-Vera, M., Motta, E., Ciravegna, F., 2006, "Semantic Annotation for Knowledge Management: Requirements and a Survey of the State of the Art", *Web Semantics*, **4**(1), pp. 14-28.

[42] 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), pp. 95-113.

 