

# Deployment of an Ontological Framework of Functional Design Knowledge

Yoshinobu Kitamura<sup>a,\*</sup>, Masakazu Kashiwase<sup>b</sup>, Masayoshi Fuse<sup>b,1</sup>, and Riichiro Mizoguchi<sup>a</sup>

<sup>a</sup> *The Institute of Scientific and Industrial Research, Osaka University,  
8-1 Mihogaoka, Ibaraki, Osaka, 567-0047 Japan*

<sup>b</sup> *Plant and Production Systems Engineering Division, Sumitomo Electric Industries, Ltd.  
1-1-1, Koya-kita, Itami, Hyogo, 664-0016 Japan*

---

## Abstract

Although the importance of knowledge sharing among designers has been widely recognized, knowledge about functionality in the conceptual design phase is hard to capture and is often scattered across technical domains. Aimed at capturing such functional knowledge that can easily be applied to other domains, we developed an ontological framework to systematically describe it. It includes six kinds of knowledge about functionality, i.e., two types of functional models, two types of organization of generic knowledge, and two ontologies of functionality. This paper reports on the successful deployment of the framework in a production company. The Plant and Production Systems Engineering Division of Sumitomo Electric Industries has used our framework to share functional design knowledge on production systems since May, 2001. An empirical evaluation by Sumitomo's engineers was unanimously positive. They said that this framework enabled them to make implicit knowledge possessed by each designer explicit and to share it among team members. This paper discusses some successful use-cases in tasks such as a design review, a patent application, and solving a quality problem. We also discuss effects of our ontological framework as a consistent viewpoint for capturing implicit functional knowledge and as a conceptual inter-lingua among designers. The limitations of our framework are also discussed.

*Keywords: Ontology, Knowledge systematization, Functional knowledge, Knowledge sharing, Design*

---

## 1. Introduction

The importance of knowledge sharing among designers and engineers has been widely recognized in knowledge-intensive engineering. Although CAD data has been shared well using the recent CAD and computer network technologies, that of knowledge about functionality has been left undeveloped. While there is no common understanding of what a function is [1]-[4], people share the idea that functional knowledge is tightly related to design intention. For example, a functional structure [5] of a product shows users how its topmost goal is achieved through sub-functions of components and sub-systems (so-called "how things work"). Such a product model from the viewpoint of functionality is called a functional model. Functional models represent a part of (but not all of) the designer's intentions, so-called design rationale [6].

As has been discussed in the research on knowledge management, making such subjective and hence implicit knowledge of a product explicit is highly needed for knowledge sharing within a community. The same applies to the design community and design knowledge sharing is expected to drastically improve the design process. For example, in activities related to

design review, an explicit description of the designer's intentions helps other people to understand the original design more effectively. Moreover, it can facilitate deeper insights into the designs by the designers themselves.

Much research has been conducted on the representation of functionality in Value Engineering [7], engineering design [4],[5],[8]-[10], and Functional Representation [11]-[20]. Practical design diagrams such as QFD (Quality Function Development), FMEA (Failure Mode and Effect Analysis) sheets, and fault trees in FTA (Fault Tree Analysis) also include functional knowledge. However, there is a gap between the theoretical and practical work being done in companies. From the experience of two of the authors who have worked in a production company, engineers have suffered from the difficulties of sharing technical (functional) knowledge for many years. They have regularly written various kinds of technical reports/documents for each of the jobs such as design review, maintenance report, reliability analysis, troubleshooting and have stored much of those in databases. Unfortunately, however, it has been difficult for them to understand documents written by other engineers and hence few such technical documents have been efficiently reused. The reasons include:

- It has been hard to describe implicit functional knowledge systematically.
- To retrieve appropriate knowledge has been hard.

---

\* Corresponding author. Tel.: +81-6-6879-8416; fax.: +81-6-6879-2123.  
Email address: kita@ei.sanken.osaka-u.ac.jp (Y.Kitamura)

<sup>1</sup> Currently, Sumitomo Wiring Systems, Ltd.

- Knowledge has been often specific to a target product or equipment.
- Representation frameworks, such as FMEA and FTA, have been task-specific.

The authors have been tackling these real problems in the industry for many years on the basis of Ontological Engineering and have established an ontological framework for functional knowledge [22]-[24]. This is based on functional ontologies, which provide viewpoints and the vocabulary for capturing functional knowledge that can be used to solve these problems (discussed in the next section).

Although a great deal of research on Ontological Engineering has been done in the last decade, little is known about its deployment in industry. This research has not focused on the generic mechanism for ontological models but on the real content of well-focused target knowledge, that is, functional knowledge that is still so generic that it could be applied to all artifacts. This framework was successfully deployed in the Production Systems Division of Sumitomo Electric Industries, Ltd.

This paper discusses use experiences and effects of the ontological framework deployed in Section 4 after a detailed analysis of the problems in Section 2 and an overview of our framework in Section 3. A knowledge management program named SOFAST<sup>®</sup> was developed for the deployment. Its architecture is described in Section 5. The success factors and limitations are analyzed in Section 6. Then, related work is discussed followed by some concluding remarks.

## 2. Ontological approach to sharing functional knowledge

A functional representation of a product consists of descriptions of the functionality of components (or (sub-)systems) and the relationship between them. Our claim is that it is not trivial to clearly identify function-related concepts and relations as we explain below. We think it is one of the deep causes of the difficulties in the industry mentioned in the Introduction.

In Value Engineering (or in similar ways in functional representation), functionality of a component is denoted as a “verb+noun” style for representing the component’s activities (or actions) and its operands (in the terminology in [10]). However, such representations cannot prevent inappropriate modeling. For example, one might describe “to weld metals” as a function of manufacturing equipment. However, “to weld” implies not only “what to achieve”, say, “to join”, but also “how to achieve” in which the metals are fused. From the functional point of view, the fusion is not regarded as a goal (“what to achieve”)

but just the method by which the goal is achieved (where fusion is the goal, its generalized goal (i.e., the super-concept in an *is-a* hierarchy) can be “to melt” or “to mix”). In fact, the same goal can be achieved with different methods (e.g., using nuts and bolts) without fusion. To allow freedom in design and to make the selection of “nut & bolt” instead of “welding” possible, the achieved function of both methods should be the same, say, “to join”. Of course, some of the characteristics of the results of joining using fusion and with “nut & bolt” are different (e.g., ease of disassembly). Such characteristics can be regarded as the conditions for selecting a method for a specific function. It is true that a functional term loses some amount of information by this information decomposition. However, what is lost is added to the information on methods. In total, functional terms can successfully be made very generic without any loss of information.

Pahl and Beitz defined some sets of a few (4-16) generally-valid functions [5]. However, they are too abstract to describe details on the designer’s intentions. In fact, there are general-specific relations (so-called *is-a*, *a-kind-of*, abstraction, or specialization relations) between functions. For example, in Hubka and Eder [10], the hierarchy for the “degree of abstraction” of functions represents the specialization of functions with additional conditions. The conditions, however, may sometimes (not always) include the characteristics of a specific method of achieving a function such as “transportation by sea” [10] which has the same difficulty as “welding”.

This difficulty in capturing functions and their relations is a special case of a general problem treated in Ontological Engineering where it is hard to distinguish the *is-a* (general-specific) relation from the *part-of* relation (so-called whole-part, micro-macro, decomposition, or aggregation relation). The *part-of* relation between functions represents how a function is achieved by finer-grained functions (we call this the “*is-achieved-by*” relation) and has been captured as function decomposition [5], whole-part relation [15], and “degree of complexity” [10]. Nevertheless, engineers are still easily confused as the above examples demonstrate.

These observations suggest the necessity of an ontological schema for functional knowledge. An ontological schema specifies not only the data structure but also the conceptual viewpoint for capturing the target world (called specification of conceptualization [25]). It provides guidelines or constraints on modeling, which helps knowledge authors describe knowledge consistently. An ontological schema for functional knowledge includes the fundamental ontology for capturing functions and the clear organization of concepts and relationships, which help the knowledge author separate “what to achieve” from “how to achieve”.

This paper mainly concentrates on such roles of ontologies in the knowledge capturing and organizing phase. The roles of ontologies in knowledge exchange and communication in engineering, on the other hand, have been investigated elsewhere [26]-[29].

In the design literature, German systematic design approach [5] has provided us with a basic viewpoint to capture functions, in which they are regarded as the input-output relations of a black-box. The black-boxes are connected and aggregated (or decomposed). In Ontological Engineering, such a device-centered viewpoint originating from systems dynamics theory is called *device ontology*. There are several examples of this [3],[30]-[33]. A device ontology is suitable as a basis for establishing ontologies of functional world, since functions are usually considered as what components or devices achieve.

We have established an ontological modeling framework, whose features include:

- An extended device ontology: Refined device ontology for capturing behaviors of components [24].
- A functional concept ontology: to provide generic functional concepts representing verbs of functions in *is-a* hierarchies [21].
- Conceptualization of “ways of function achievement” and their *is-a* hierarchy for detaching them from functions [22].
- Four types of functional knowledge and ontological modeling guidelines [23].
- Integration of information on unintended use for maintenance [34],[35].

Such ontological commitments help designers explicate their own thoughts on the design and share it with a design team. This paper discusses such effects in the deployment in Section 4. The last feature is to solve the last problem (i.e., task specific representation

such as FMEA) mentioned in the Introduction.

### 3. Ontological Modeling Framework

Our framework for functional-knowledge modeling is described in Fig. 1. This framework is an extension of our functional modeling language FBRL (abbreviation of a Function and Behavior Representation Language) [36]. It shows the modeling process from the functional model of a concrete artifact to well-organized generic knowledge. It includes six kinds of knowledge about functionality.

This modeling framework is based on an extended device ontology (Fig. 1(f)) as a basis for conceptualizing the functional world. We extended the conventional one mentioned in the previous section by redefining the concepts of “conduit” and “medium” to provide knowledge authors better ontological guidance and to cope with mechanical domains that seemingly do not fit the device ontology [24],[37]. The extended device ontology specifies “roles” played by physical things (“participants”) taking part in the physical world.

Based on the extended device ontology, the “behavior” of a device is defined as the objective (independent of designer’s intentions) interpretation of its input-output relations considering the device as a black box. The description of the behavior is independent of the system (i.e. context) in which it is embedded. A device is connected to one another through its input or output ports. A device plays a role as an “agent”, which changes the states of things being input (called “operand”, i.e., the thing being processed by the device) such as fluid, energy, motion, force, and information. The input-output relation of the behavior is, more precisely, the difference between the states of the operand at the input port and that at the output port. A device can be a mechanical element, a mechanical pair, a component, an assembly, a

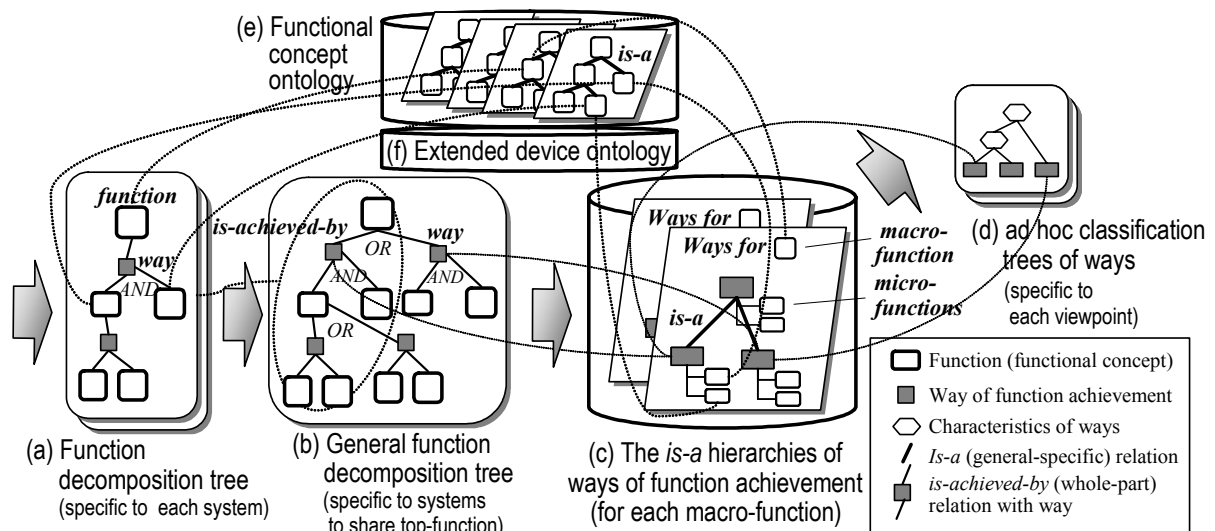


Figure 1. Framework for functional-knowledge modeling.

sub-system, or a system. These include both products and manufacturing machines.

A “conduit” is defined as a special type of device that can be considered as transmitting an operand to the output port without any change in the ideal situation. Examples include a pipe for liquid and a shaft for torque. One of the reasons for having the conduit concept is that we can neglect the “to transmit” function common to all devices that transmit input things to the output. A “medium” is something that holds an operand and enables it to flow among devices. One example is steam for heat energy. In some domains, the thing playing the conduit role can play the role of medium as well. For example, while a shaft is a conduit for force and motion, at the same time, it is also the medium for them.

A function is defined as the teleological interpretation of a behavior under an intended goal [21],[36]. Although the selection of a function (i.e., interpretation of behavior) is dependent on the (sub-)system in which it is embedded, the definitions of functions themselves can be done locally. Such definition of function which distinguishes the functional interpretation from the micro-macro relation is different from those in [12],[17] which are based on the micro-macro relationship. The definitions of functions in the literature [10],[15],[16] also distinguish them. However, we explicate mapping primitives between behavior and function (called functional toppings (FTs) [36]) and the operational conceptualization of functional concepts.

We developed an ontology of generic functions (called functional concept ontology) shown in Fig. 1(e) with such operational definitions [21]. It defines about 220 concepts in four kinds of *is-a* hierarchies.

On the basis of these two ontologies, a function decomposition tree (Fig. 1(a) and examples are in Figs. 2 and 3) first models the functional structure of a specific device. All functions (rounded box nodes in the tree) in the function decomposition tree are instances of generic functions defined in the functional concept ontology. This basically represents that a required function (called a macro-function) can be achieved by the sequence of specific sub(micro)-functions. This relation is a kind of “part-of” relations or aggregation relations between functions.

In this framework, “the reason why a function can be achieved” is conceptualized as a “*way of function achievement*”. It explicates background knowledge on functional decomposition such as physical principles and theories. In Figs. 1(a) and 2, the *way* is represented by the small dark squares that connect the whole function to sub-functions. Moreover, the framework includes unintended behaviors [34],[35]. It is needed for explicating the design rationale for supplementary functions to prevent them from

occurring. It is important in design review activities and equipment improvements as discussed in Section 4.

Second, a *general function decomposition tree* (b) is composed of some function decomposition trees for similar devices with the same whole-function. It includes alternative *ways of function achievement* in an OR relationship. It represents possible *ways* to achieve a specific function. Fig. 4 shows an example.

Last, a concrete *way of function achievement* in a (general) function decomposition tree is generalized into a generic *way of function achievement* (called functional way knowledge). Then, *ways* to achieve the same function are organized in *is-a* relations according to their principles (called an *is-a hierarchy of ways of function achievement* (c) and some examples are shown in Fig. 5). We distinguish the organization as an *is-a* hierarchy from the other derivative organizations depending on viewpoints (called an *ad hoc classification tree* (d)). The ad hoc classification trees can be reorganized by a *functional way server* according to a given viewpoint [22]. Such generic functional knowledge is somewhat similar to that presented in [8],[16],[17]. We will discuss the differences in Section 7.

Of most importance in this conceptualization is that these types of trees concerning functions in Fig. 1 are different from one another despite the superficial similarity. The function decomposition tree (a) represents *is-achieved-by* (a kind of *part-of*) relations between functions. The *is-a* hierarchies of *ways of function achievement* (c) represent abstractions of key information about how to achieve the function, while the *is-a* hierarchies in the functional concept ontology (e) represent abstractions of functions themselves, i.e., what to achieve. Moreover, the number of *ways* to achieve a function is huge in nature, while the number of functional concepts is small.

Currently, the guidelines for building these trees are being developed and are concerned with agents and operands of functions, relations between sub-functions, and “is-achieved-by” relations [23]. They help a modeler capture functional structures based on the extended device ontology. For example, one of the guidelines prescribes that sub-functions must contribute to achieving the macro-function clearly on the basis of the physical principles represented as the *way of function achievement*. According to this guideline, a modeler should check for the existence of implicit functions.

## 4. Deployment

The ontology and the modeling framework for functional knowledge have been deployed for over three years at the Plant and Production Systems Engineering Division of Sumitomo Electric Industries,

Ltd. (hereinafter referred to as SEI). The purpose was to share functional knowledge among engineers in the division about the production facilities used in their daily work.

After a one-year study of our framework, test use was commenced in February 2001. In May, 2001, use on real problems encountered in daily work was started. A knowledge management software named SOFAST (discussed in the next section) has been deployed since December, 2002. Currently, about 50 people in three factories use the framework in their daily duties. The database for the SOFAST software includes 103 (generic) function decomposition trees for machines used in real work. The targets are manufacturing facilities that are mainly used in semiconductor manufacturing processes including machines to slice semiconductor ingots (wire-saw and inner-blade types), machines to polish wafers, a tension control system, a machine to adjust optical fiber connections and inspection machines.

First, function decomposition trees were described to share understanding of the target facilities as discussed in Section 4.1 and Fig. 2. They were also used for improving facilities as discussed in Section 4.2 and Fig. 3. Next, general function decomposition trees were used for the design review (Section 4.3) and patent application (Section 4.4 and Fig. 4). Such function decomposition trees could be shared with different types of engineers and in different tasks as discussed in Section 4.5. Last, specific ways of function achievement in function decomposition trees were generalized and organized in *is-a* hierarchies. Such kinds of knowledge base can be used to explore ways of doing conceptual design as discussed in Section 4.6.

A users' group of SOFAST software for companies was established in April, 2003. There are currently 13 member companies where test use has been done.

#### 4.1 Understanding and Sharing DRs

Figure 2 shows a function decomposition tree of a production machine called a wire-saw. It has been designed to slice semiconductor ingots with moving wires. The top function is "to split" rather than "to slice", since "to slice" implies how to split and specific information about the thinness of the split part. The former information is regarded as a *way of function achievement*. The latter information is regarded as the quantitative degree of results of a function. Such specific information about the degree of a function can be used as the conditions for selecting a *way of function achievement* from the available ways for achieving the function. Splitting is achieved with two sub-functions; losing the combination force of the part (kerf loss, i.e., part lost by cutting) and moving the part away. This *way of function achievement* is conceptualized as the "removing way" based on the separation of the kerf loss part. The sub-functions are further decomposed into smaller sub-functions.

Figure 2 includes possible unintended behaviors of the device (phenomena) and supplementary functions to avoid them. For example, the cooling function for the moving wire has been designed to keep the wire from snapping due to heat caused by friction. The relation between wire snapping as a possible unintended phenomena (or trouble) and frictional heat as its cause is explicitly described. In other words, our framework provides knowledge media on such designer's intentions.

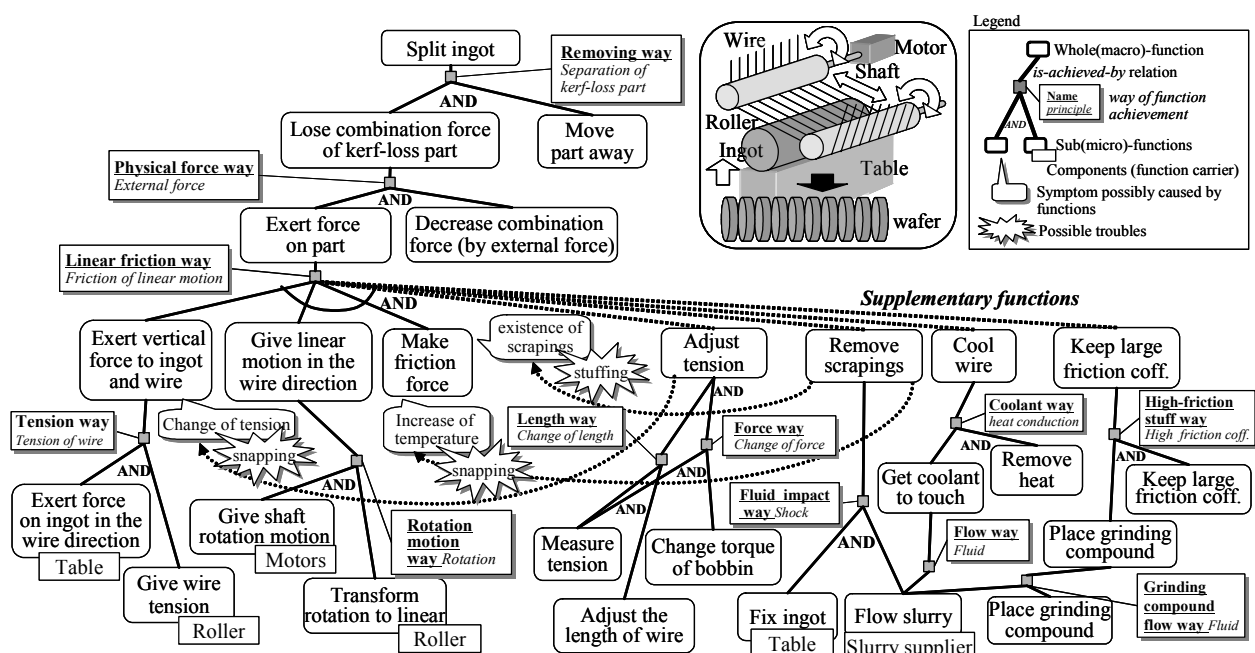


Figure 2. Function decomposition tree of a wire-saw for slicing ingots (portion).

Such a function decomposition tree shows the designer's intentions on how to achieve the goal, which is not included in either the structural or behavioral model. This effect is basically the same as in the conventional function decomposition tree [5]. The main distinctive features of our framework include (1) the concept of "way of function achievement" and its relationship with functions, (2) the extended device ontology, and (3) integration of unintended behaviors explained as follows. First, the concept of "way of function achievement" and definitions of relationships with functions discussed in Section 3 help knowledge authors to keep functions representing "what to achieve". For example, as mentioned in Section 2, the pseudo function "to weld" can be decomposed into the "joining function" and the "fusion way of function achievement". The characteristics of welding such as the melting of objects are described as properties of the *fusion way*. To use the *way of function achievement* is not mandatory, since it can be left anonymous when there is no necessity to conceptualize it. Therefore, its introduction is not restrictive in building a function decomposition tree.

Second, the extended device ontology provides concepts for assigning "roles" for each object in the target world. In Fig. 2, the wire can be considered as an *agent* (exerting force on ingots), an *operand* (moved by the roller) or a *conduit* (transmitting tension). According to semantic constraints in the extended device ontology, a possible way to consistently assign roles is to decompose the wire into two parts, a working wire as an agent and a transmitting wire as both the medium (a sub-concept of the operand) and conduit. One extension to the device ontology is to accept the last situation.

Last, the integration of unintended behaviors into the function decomposition tree enables us to understand

the intention of the supplementary functions, which often give important information. For example, another designer can understand that the reason for the existence of the supplementary function "to cool wire" in Fig. 2 is to avoid snapping by removing the cause, i.e., frictional heat.

The experiential evaluation on this aspect by the SEI engineers was unanimously positive. They said that this framework enabled them to explicate implicit knowledge possessed by all designers and to share it among team members. It was easy for designers to become familiar with the framework based on the device ontology. They said that the explication of their own implicit design knowledge helped them reflect on the design and/or the target devices. This benefit provided them with strong motivation for describing the functional models.

## 4.2 Equipment Improvement

This section reports on a real example of making implicit knowledge about the functionality of the manufacturing equipment explicit and the improvement resulting from the explicated knowledge. The target manufacturing equipment was a machine for polishing semiconductor wafers. As we can see from Fig. 3, a weighted rotating disk polishes the wafer on a table with slurry containing diamond powder as the grinding ingredient. The rotating disk moved freely inside an outer ring called the guide ring. The goal for improvement was to reduce the time necessary to grind a wafer to 63%. To do this, an engineer initially tried to adjust the values of the working parameters of the machine such as the rotating speed of the disk and the weight and amount of slurry. After four months of investigations, however, the results were still not conclusive to reach this goal.

Then, to establish another (unknown) working parameter, he described the function decomposition

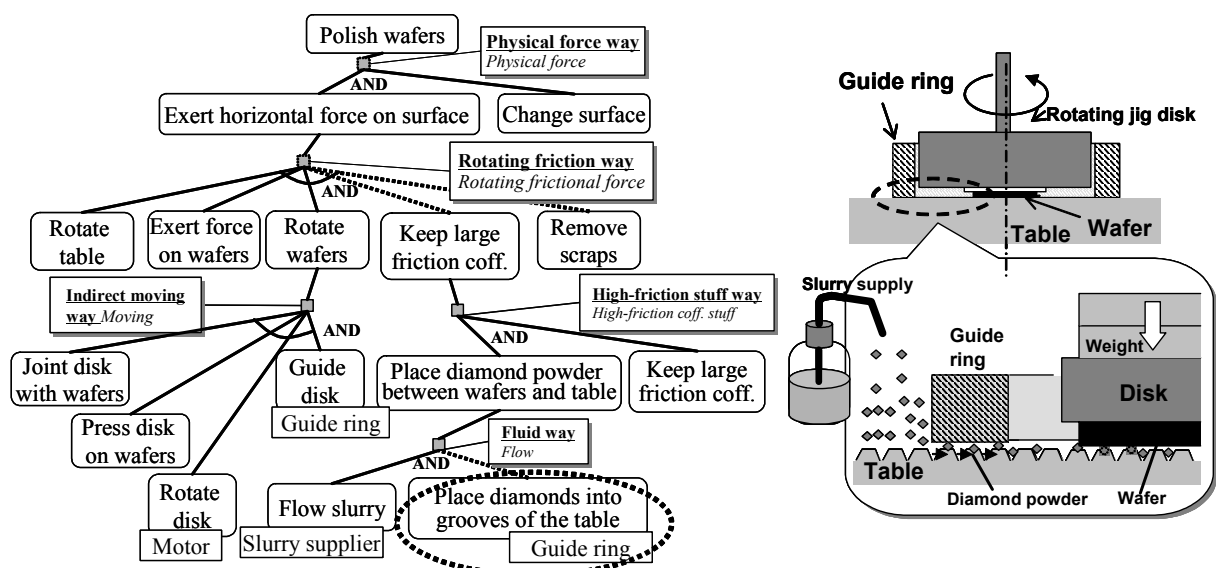


Figure 3. Function decomposition tree of a polisher to find parameters

tree of the machine (Fig. 3). He referred to another function decomposition tree of the wire-saw (Fig. 2), which had been stored in the database discussed in Section 5. Although these two machines had different main functions, he found a shared function “to maintain a large friction coefficient” and its sub-function “to place diamond powder between the wafers and the table” by referring to the function “to place grinding compound” in the function decomposition tree of the wire-saw. This reused intermediate sub-function clarified a part of the mechanism required to achieve the high time efficiency. As a result, he became aware of the additional (and implicit) function of the guide ring, i.e., to place the diamond powder in the slurry into the grooves of the table to achieve the function “to place diamond powder between the wafers and the table”. Thus, he changed the width of the guide ring so that more diamond powder could be placed into the gaps between the wafer and the table. Eventually, the necessary time was reduced to 76%, which was better than the initial goal. This improvement was achieved within three weeks.

### 4.3 Design Review

Design review is a team activity to double-check the original design and explore possible alternatives. To explain the original design, the SEI engineers had been using a comparative table with a text, which described alternative designs in columns with their features in rows.

The Production Systems Division of SEI adopted a general function decomposition tree as the regular schema for design-review documents. It showed alternative ways of achieving functions for each (sub) function, their features in comparison, and reasons for adopting a specific way, or not, in the one figure. It was difficult to describe all alternatives exhaustively for each function in the comparative table. Thus, the design review had to be done thrice on average. After adopting our framework, however, the number of

times the design reviews had to be done was reduced to one third.

The description of unintended behaviors and supplementary functions played an especially crucial role in design for reliability. It showed the original design took into account what phenomena could possibly occur and how to avoid these with additional supplementary functions (Fig. 2).

### 4.4 Patent Map and Patent Applications

Another use for the general function decomposition tree is as a kind of “patent map” to indicate applicable ways to achieve a function. For example, Figure 4 shows a general function decomposition tree including patented ways of handling wafers. The italics are abbreviated names of companies which adopt each way for manufacturing wafers. The differences in working principles and features of patents are organized in each level of function decomposition. It includes possible unintended (undesirable) phenomena and problems such as wafer chipping for each way. Differences (originality) in the new patent from conventional ones can be explicitly described in patent applications as new ways of function achievement and/or new features of ways.

In applying for a new patent, it is difficult for engineers and patent attorneys to clarify its originality and to make the proper claims. At SEI, it took a lot of hard effort over a long period, on average, three or four weeks. When a patent application was completed with the general function decomposition tree in Fig. 4 (plus new ways of function achievement to create new patent), the period was reduced to just one week (i.e., one third the usual time). Moreover, the patent claims were increased, in some cases doubled. This was because the patent attorney found additional differences with other patents by checking each level of function decomposition in the general function decomposition tree.

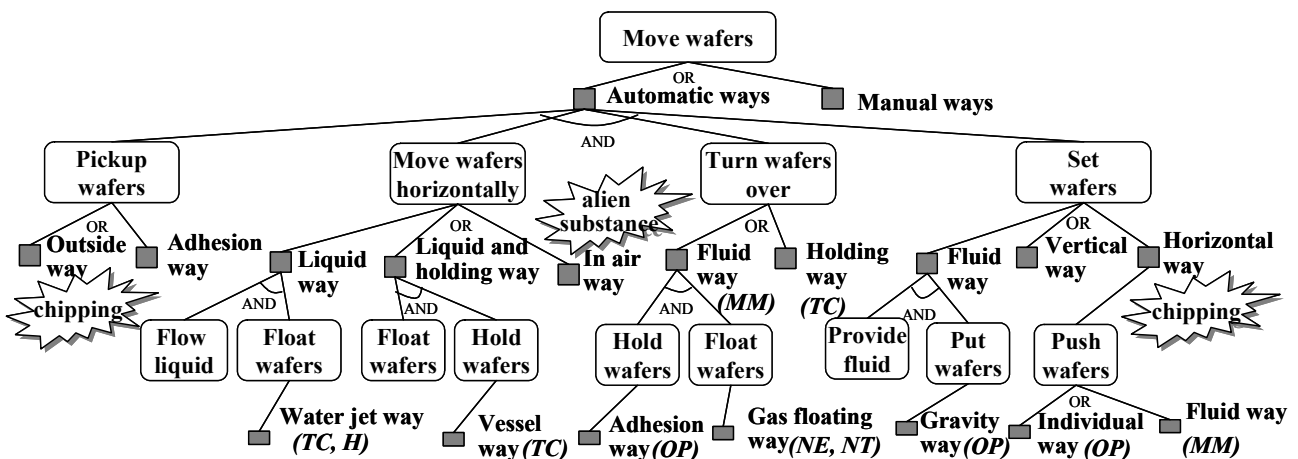


Figure 4. A patent survey for handling semiconductor wafers

#### 4.5 Sharing with Different Types of Engineers and Sharing in Different Tasks

In addition to designers and patent attorneys using our framework as knowledge media for communication, designers, manufacturing engineers, manufacturing equipment engineers, equipment operators, and equipment maintenance personnel, all used it in collaborative work. Although mutual understanding and collaboration are urgently required, this rarely happens because they have their own viewpoints and use different knowledge representations such as design drawings, QFD, FMEA sheets, and FTA diagrams. The problem is in that each representation uses its own vocabulary and lacks interoperability with the others. The use of our framework, however, facilitated their mutual understanding and collaboration in a project to improve equipment where it worked as a common vocabulary, which was lacking before.

Another kind of interoperability of knowledge in our framework was among different tasks (engineering activities) such as design, solving quality problems, and patent applications. For example, the function decomposition tree described to review designs (as discussed in Section 4.3) was reused to diagnose a problem with equipment deployed on the manufacturing line and to improve the equipment. It was derived from a sub-system that adjusted the angle at which ingots were cut with the wire saw (Fig. 2). A new FTA diagram might have been described to solve the problem in conventional work. Moreover, we applied a new patent to adjust the sub-system using the same function decomposition tree plus information from existing patents that were related. This means that the same knowledge representation was reused in different tasks, i.e., design, diagnosis, improvements, and patent applications, which was impossible before at SEI.

#### 4.6 Expanding Alternative Ways of Function Achievement using Generic Ways

The specific *ways of function achievement* are generalized into generic *ways* and then organized in *is-a* hierarchies. Figure 5 shows *is-a* hierarchies for *ways of function achievement* for split functions and others. They have been generalized from specific *ways* used with the wire-saw (Fig. 2) and other cutting machines such as water jets and electrolysis. Within this organization, the differences between the wire-saw and other cutting machines are explicitly represented. Wire-sawing uses three *ways*, i.e., the *removing way* for splitting, the *physical force way* for losing combination force, and the *linear friction way* for exerting force. Moreover, the ways of exerting force can also be used for other machines, e.g., washing machines. For example, dirt is separated from clothes by random friction force caused by the rotating screw in a screw-type washing machine. This suggests that these pieces of knowledge are general and can be applied to different domains. The conventional organization for *ways* of cutting found in text books in the field does not mainly show principles but “what something is used for” such as wires and blades. “The wire-saw way” found in text books is not a single *way of function achievement* but a composite made up of three primitive ways.

Although knowledge on such organization of *ways of function achievement* has not been fully deployed yet, a feasible new improvement to wire-sawing was found in the knowledge-base, i.e., we manually found a *way* of using the magnetic fluid that is used in the textile industry to control the tension of the wire. This simple invention could have been done by a computer system called the functional way server [22]. This indicates the utility of our framework for general functional knowledge.

Techniques of shedding light used in inspection machines were also systematized in deployment. Consulting systematized generic *ways* in the

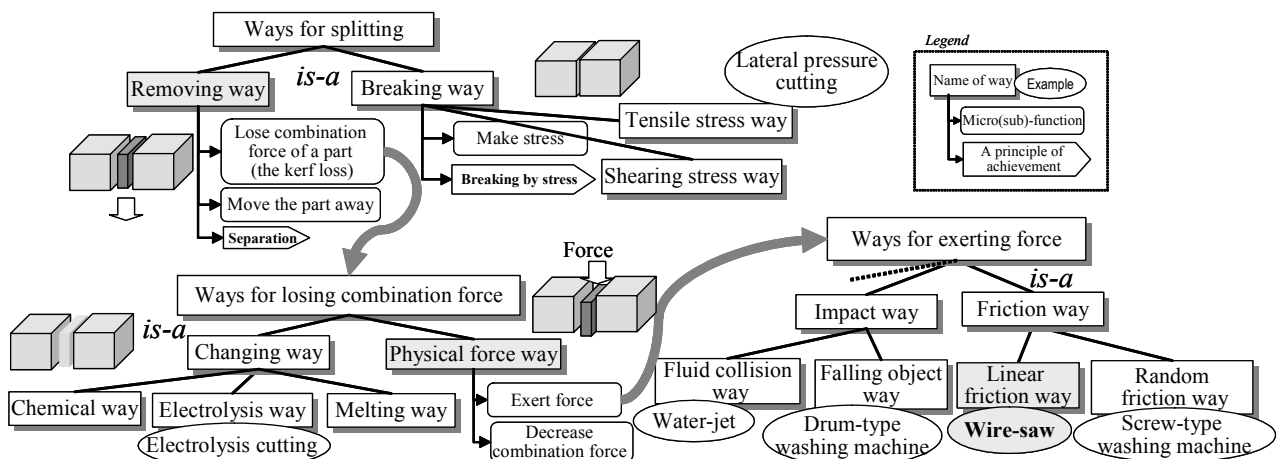


Figure 5. An example of organizing generic ways in *is-a* hierarchies

knowledge-base, a novice engineer developed an inspection machine in three days. Experts usually need two weeks for such developments.

## 5. SOFAST<sup>®</sup> software

We developed a knowledge management software named SOFAST<sup>®</sup> (abbreviation for Sumitomo Osaka-university Function Analysis and Systematization Tool and registered trademark of SEI.), which was designed to support the description of functional knowledge and sharing in an intra-network. It consists of client software and an SQL server (Fig. 6(a)). Using the client software, a user can describe function decomposition trees on a graphical user-interface. Figure 6(b) has a screen snapshot, where both main panes display a general function decomposition tree in different styles in Japanese. As we can see from the right pane, users can attach related documents including bitmap images, graphs, and spreadsheets to the tree. The small window in the middle is an overview of the whole tree for scrolling.

The described models are stored in the SQL server and can be accessed by users with the client software (or any SQL-support software) from other hosts. Because the server stores each *way of function achievement* at each level separately, a user can retrieve many *ways of function achievement* from different facilities or products to achieve the specified function by specifying a goal function. For example, 56 instances of *ways of function achievement* for “to shed light” from many facilities (including different applications of the same generic *way*) can be found from the current SOFAST database, which includes the 103 (general or specific) function decomposition trees.

Since April, 2003, we have provided SOFAST to 13

other companies in the SOFAST users’ group. The authors regularly hold meetings with the member-companies for lectures, training, reports on use, and discussions on further improvements.

The current implementation of SOFAST is, as discussed in detail in Section 6.2, data storage software rather than an intelligent design support system. The success of the deployment discussed thus far is accomplished not only by the functionality of the SOFAST software but also by the lectures and training in the collaborative work done between Osaka University, SEI, and the users’ group. The lectures and training are aimed at reinforcing the constraints of the ontologies discussed in Section 3 on knowledge authors. As we will discuss in the next section, the factors for success mainly result from such ontological commitments. Moreover, the management capability of *is-a* hierarchies of generic *ways of function achievement* is missing. The built-in vocabulary of functional concepts is not based on the functional concept ontology. Improvement in this respect is discussed in Section 6.2.

## 6. Discussion

### 6.1 Success Factors for Deployment

The successful deployment discussed thus far is a kind of knowledge management activity. In general, difficulties with knowledge management activities include:

- Difficulty in explicating implicit knowledge,
- Difficulty in retrieving useful knowledge, and
- Lack of motivation in writing own knowledge.

First, it is difficult to explicate one’s own implicit knowledge. Functional knowledge is intrinsically subjective, not objective. Without guideline, novice

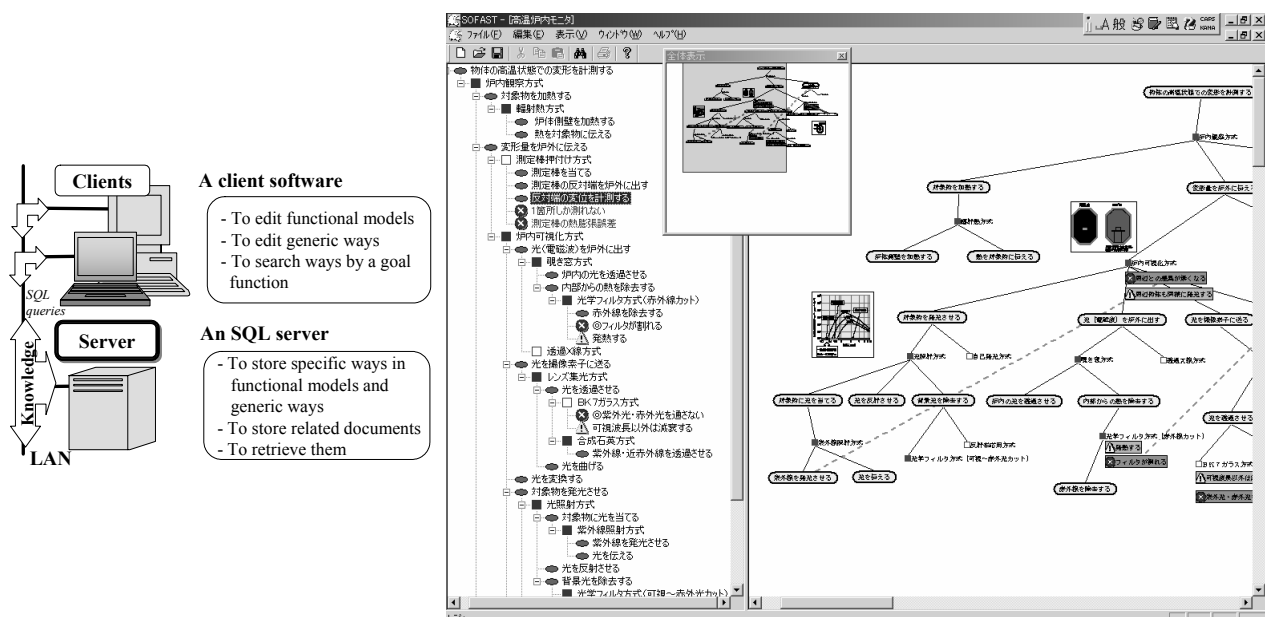


Figure 6. Knowledge management software SOFAST.  
(a) Architecture (left) and (b) screen snapshot of client software (right)

modelers would be puzzled in describing functional models. Functional ontologies provide conceptual rules or guidelines to capture the target world, i.e., conceptual and subjective functions. Extended device ontologies, especially, provide users with hints on interpreting how a device works consistently as discussed in Section 4.1. The concept of “way of function achievement” also helps modelers to eliminate the confusion between “what to achieve” and “how to achieve it”. A clear distinction between a general-specific hierarchy (*is-a* relations) and a whole-part hierarchy (*is-achieved-by* relations) helps knowledge authors to have consistent descriptions of function decomposition trees and *is-a* hierarchies of *ways of function achievement*. This avoids the confusion between the two, which has often occurred.

Although difficulty in retrieval is sometimes treated as a technical issue in information searches, we believe that the retrieval problem with functional knowledge is due to the dependence of content on “how to achieve functions” and “how to use information”. When functional terms used are composites of “how to achieve” and “what to achieve” like they were before (e.g., “to weld”), the terms are very domain- and equipment-dependent and hence generality is low. This has caused the low retrievability of knowledge. This issue can be avoided by the concept of “way of function achievement” as previously discussed because its detachment from functional concepts makes the functional terms very general.

“How to use information” is also important in the knowledge management context. The knowledge found has to be ready for use in the task at hand. To do this, well-prepared knowledge representation needs to be available which the knowledge should fit. This issue is partially resolved by our functional ontology framework which is an integration of knowledge about functional structures representing intended behaviors and unintended behaviors at an appropriate level of abstraction, i.e., the functional level. This enables the same model (representation) to be used in different tasks as discussed in Section 4.5. Such applicability in multiple tasks reduces the effort required by knowledge authors. Furthermore, it augments interoperability among task-dependent knowledge representation via our framework.

Last, in general, knowledge authors have no effective motivation to write their own knowledge and share it with others. In deployment, however, engineers themselves say that they obtain benefits from writing functional models of their own equipment, since it gives them the chance to reflect and obtain good stimuli, which leads them to an in-depth understanding of the equipment. This is enabled by our modeling framework operating as a knowledge medium, which externalizes the engineers’ understanding, which had been implicit, to an appropriate level of abstraction with consistent guidance. There was a real example in

Section 4.2. In general, the micro-macro hierarchy of the function decomposition tree enables the designer to systematically explore possible alternatives (for conceptual design) and/or causes of the problem (for problem solving) for each function. For example, fault tree analysis (FTA) for problem solving tends to make it difficult to enumerate all possible causes without a clear understanding of the function structure of the target device.

## 6.2 Limitations

The main point of our framework is that it adopts an ontological approach to controlling the content of functional models. However, it has some disadvantages. The first one is less freedom of functional representation than ad hoc functional modeling. It became a problem especially in selecting a functional concept for a component. There could be a domain-specific vocabulary and different terms for the same concept. We cannot claim completeness of concepts in our functional concept ontology due to its nature and understand the necessity of extending it. In order to make the use of SOFAST easier, the latest version supports multi-level terms which consist of the functional concepts (defined in the ontology), *usual function words* and domain-specific vocabulary. The *usual function words* are verbs for representing functions appearing in daily work, and have been prepared beforehand by collaboration with companies in the users’ group. Such words are associated with each functional concept. SOFAST of the latest version thus allows knowledge authors to use terms rather freely.

The second disadvantage of our ontological approach is that it needs training for writing functional models that are compliant with functional ontologies. In other words, it is not very easy to impose ontological commitments on knowledge authors. The authors are currently establishing stepwise guidelines for describing functional knowledge to enable easier commitment to the ontologies [23]. Moreover, automatic checking of violations in the functional models against ontologies is being investigated. Neither of our ontologies is just a taxonomy. Their definitions are structured with slots and constraints and include axioms as a result of deep insights into the behaviors and functions in physical systems [22]. Such formal definitions can be used to automatically check the models.

As a result of these limitations, some of the functional models described thus far do not follow the functional ontologies completely in deployment. The current vocabulary used in SOFAST is not fully based on the functional concept ontology. The main use of SOFAST, currently, is to describe the function decomposition trees of each production facility and a general function decomposition tree for similar facilities. Sharing *ways of function achievement* in

SOFAST does not rely on the *is-a* hierarchy of generic *ways* but on searching for specific *ways* of function achievement by specifying the goal function. Nevertheless, discrimination of *is-a* relations from *ways of function achievement* has helped engineers avoid a great deal of confusion. Such advanced features of our framework are to be deployed by implementing new functionalities to support them in SOFAST together with advanced training.

Apart from the production systems and facilities discussed in this paper, ontologies have been applied to modelling a power plant [21], an oil refinery plant, a chemical plant, a washing machine, a printing device, and manufacturing processes. The models have taken into account changes in thermal energy, flow rate, and ingredients of fluids, including force and motion of operands. The current functional concept ontology can describe simple mechanical products, although it does not cover static force balancing and complex mechanical phenomena based on the shape of operands. The modelling framework currently cannot cope with the human philological process, body movements (so-called *therblig* in Industrial Engineering), business processes, or software processes.

## 7. Related work

### 7.1 Engineering Ontologies

Of the types of ontologies, we did not concentrate on the task ontology of design activities (such as Chandrasekaran [38]) but the domain ontology of artifacts to be designed. Much work on ontologies in the engineering domain [3],[26]-[33],[39]-[41] has been done. One remarkable example is by Borst, et al.[32], in which the PhysSys ontology was proposed as a sophisticated lattice of ontologies for the engineering domain. However, the device ontology in PhysSys is weak in the same way as conventional ones in that it does not have sufficient concepts to enable the *ontological roles* all participants play to be understood. We extended such conventional one by redefining “conduit” and “medium” as an extended device ontology. Our refinement to the device ontology provides the modeller with more detailed guidelines to capture target devices.

Moreover, the PhysSys ontology has no ontology for functions from the teleological viewpoint. Chandrasekaran and Josephson clarified meanings of the concept “function” based on ontological considerations and proposed two types of functions, i.e., device-centric and environment-centric [3]. Although we share this distinction and their attitude towards ontological analysis, we only concentrated on a device-centric viewpoint in this paper. Other classifications of functions can be found in [2],[4],[20].

Other ontological considerations on functionality can be found in [33],[40].

Recently, ontologies have played a crucial role on the emerging Semantic Web in giving interoperability to web resources (e.g., [42]). Knowledge on unintended behaviors discussed in this paper can be regarded as different knowledge resources based on ontologies that are different from the functional ontology. The authors are currently investigating design knowledge transformation based on such multiple ontologies on the Semantic Web.

### 7.2 Functional representation

There has been a great deal of research on functional representation [3],[8],[11]-[20],[43]. We did not focus on the purpose function but on the technical functions in the terminology in [4],[10].

The main point of our research is to clarify several relationships related to functionality, i.e., the *is-a* hierarchy of functions, the *is-achieved-by (part-of)* relations among functions, and the *is-a* hierarchy of *ways of function achievement*. We detach a part of the conditions for specialization in [4],[10] (see discussion in Section 2) and thus described them as specific attributes of the *way of function achievement* in an *is-a* hierarchy [22],[23].

Similarly to the *way of function achievement*, a feature of function decomposition can also be found as a “means” in [18],[19],[44]. In [18], it is not generic knowledge but a model specific to a product. In [19], although generic knowledge of single functional decomposition is discussed as “means”, organization between them is not discussed. We defined *is-a* relations between conceptualized generic *ways of function achievement*, and investigated how to organize them.

The design prototypes [8] include structural decomposition as well as function decomposition. In the FBS modeling framework [16], the function prototype includes physical features of behavior to achieve the function as well as generic function decomposition. Our description of *ways* tried to maximize its generality by pointing out partial (and abstract) information on structure and behavior.

Patterns of function achievement or so-called design catalogs can be found in the design literature [5]. However, they mainly concentrate on concrete mechanical pairs.

We defined functional concepts using operational information called functional toppings (FTs) such as those that focus on operands and the necessity for operands, which then enable us to define intention-rich functional concepts [21]. Many “verb+noun”-style functional representations lack such operability. For example, standard sets of verbs (i.e., functional

concepts) proposed for value analysis [45] have no machine understandable definition of concepts. Although the recent efforts for a standard taxonomy for engineering functions by the NIST Design Repository Project [46] are well established, they lack operational relationship with behaviors.

De Kleer defines function as a causal pattern between variables [11]. In the FBS model [16], the functional symbol in natural language in the verb+noun style represents the intention of designers. We tried to identify operational primitives as FTs to represent intentions. Keuneke defines types of functions such as ToMake [13]. Our FTs include these. Although we did not adopt either the process ontology [39] or bond graph theory [47], our functional concept ontology includes similar functions in flow-based functional modeling approaches [14],[15].

The teleological interpretation specified by FTs in our approach is similar to the “means and ends” [15], F-B relationship [16], and “aims-means” [10]. The last axis includes design requirements as well [10]. Gero [48] coped with dynamic changes in the design context such as requirements.

Andreasen et al. identified several structures not only including the “functional oriented structure” but also the “product life oriented structure” for so-called DFX: Design for “something” [9].

In IDEAL [17], generic teleological mechanisms (GTM) are used (modified) to design different contexts based on analogies. In our approach, based on a limited set of functional concepts, designers can explore explicit *is-a* hierarchies of *ways of function achievement*.

TRIZ (TIPS) theory provides some patterns (or strategies) for inventions based on the contradiction between two physical quantities [49]. 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.

### 7.3 Unintended behaviors

Information on unintended behaviors in our models can be found in other formalisms such as FMEA sheets and fault trees in FTA. One of the benefits of our framework is the tight integration of such information with functional structures. This integration helps designers to systematically explore possible causes for each function at each grain-size.

To capture a larger set of failure modes systematically, research has been done on advanced FMEA (e.g. [50],[51]) using behavioral models to simulate device behaviors. The model-based diagnosis community uses sophisticated qualitative reasoning to identify

faulty components (e.g. [52]). Diagnosis using hierarchical functional models instead of behavior models have been proposed [14],[53]. These approaches based on deviations from “intended” behavioral models, however, cannot deal with parts of causative chains of faults as was pointed out in [54-56].

## 8. Concluding remarks

The successful deployment of an ontological modeling schema for functional knowledge has been reported. After we discussed the current problems in industry and expected roles of ontologies, six types of real usages in deployment, effects of ontologies, and their success factors were discussed. One of the main success factors was clarifying three types of relationships related to functionality (i.e., the *is-a* relation of functions, *is-achieved-by (part-of)* relations between functions, and the *is-a* relation of *ways of function achievement*) and to provide four types of schemata for functional knowledge. Although engineers mainly use (general) function decomposition trees in deployment, such discrimination helps engineers reflect on their own knowledge.

As discussed in Section 6.2, support for deeper commitment on functional ontologies is being investigated. The authors are planning to develop support functionality in SOFAST and to deploy it in daily work. The other 13 companies in the SOFAST users’ group will help us to improve the software.

The modeling schema in this article for unintended behaviors was a simplified version. We are currently investigating more detailed ontological schema aiming at explicit representations of design rationales for supplementary functions [35]. In our collaborative work with the Delft University of Technology, we are extending the framework to include user actions as well [34]. Interoperability between different tasks is limited in the sense that the same representation is used for all tasks. Dynamic transformation of the representation of such models is being investigated.

The authors believe that Ontology Engineering has contributed to the systematization of domain knowledge by providing a “theory of content” for knowledge, i.e., how to capture knowledge and to organize it so that it can be applied to other contexts [37]. Ontology provides fundamental guidelines for capturing the target world and for describing it in computer systems. This research can be regarded as a successful example of this research direction.

### Acknowledgements

The authors would like to thank Yusuke Koji, Mariko Yoshikawa, Tomonobu Takahashi, Masaru Takahashi, and Kouji Kozaki for their contributions to this work.

Special thanks go to 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.

## Reference

- [1] Umeda Y, Tomiyama T. Functional Reasoning in Design, IEEE Exert 1997;March/April:42-8.
- [2] Chittaro L, Kumar AN. Reasoning about Function and its Applications to Engineering, Artificial Intelligence in Engineering 1998;12:331-6.
- [3] Chandrasekaran B, Josephson JR. Function in Device Representation, Engineering with Computers 2000;16(3/4):162-77.
- [4] Hubka V, Eder WE. Functions Revisited. In Proc. of ICED 01 2001.
- [5] Pahl G, Beitz W. Engineering design - a systematic approach. The Design Council; 1988.
- [6] Chandrasekaran B, Goel AK, and Iwasaki Y. Functional representation as design rationale. Computer 1993:48-56.
- [7] Miles LD. Techniques of value analysis and engineering. McGraw-hill; 1961.
- [8] Gero JS. Design Prototypes: A Knowledge Representation Schema for Design. AI Magazine. 1990;11(4):26-36.
- [9] Andreasen MM, Hansen CT, Mortensen NH. The Structuring of Products and Product Programmes. In Proc. of the 2nd WDK Workshop on Product Structuring 1996;15-43.
- [10] Hubka V, Eder WE. Theory of Technical Systems. Berlin: Springer-Verlag; 1998.
- [11] De Kleer J. How Circuits Work. Artificial Intelligence 1984;24:205-80.
- [12] Sembugamoorthy V, Chandrasekaran B. Functional representation of devices and compilation of diagnostic problem-solving systems. In Experience, memory and Reasoning 1986:47-73.
- [13] Keuneke AM. A. Device Representation: the Significance of Functional Knowledge. IEEE Expert 1991;24:22-5.
- [14] Chittaro L, Guida G, Tasso C, Toppano E. Functional and Teleological Knowledge in the Multi-Modeling Approach for Reasoning about Physical Systems: A Case Study in Diagnosis. IEEE Transactions on Systems, Man, and Cybernetics 1993;23(6):1718-51.
- [15] Lind M. Modeling Goals and Functions of Complex Industrial Plants. Applied artificial intelligence 1994;8:259-83.
- [16] 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 1996;10:275-88.
- [17] Bhatta SR, Goel AK. A Functional Theory of Design Patterns. In Proc. of IJCAI-97 1997:294-300.
- [18] Malmqvist J. Improved function-means trees by inclusion of design history information. Journal of Engineering Design 1997;8(2):107-17.
- [19] Bracewell RH, Wallace KM. Designing a Representation to Support Function-Means based Synthesis of Mechanical Design Solutions. In Proc of ICED 01. 2001.
- [20] Deng YM. Function and Behavior representation in conceptual mechanical design. Artificial Intelligence for Engineering Design, Analysis and Manufacturing 2002;16:343-62.
- [21] Kitamura Y, Sano T, Namba K, Mizoguchi R. A Functional Concept Ontology and Its Application to Automatic Identification of Functional Structures. Advanced Engineering Informatics 2002;16(2):145-63.
- [22] Kitamura Y, Mizoguchi R. Ontology-based description of functional design knowledge and its use in a functional way server. Expert Systems with Application 2003;24(2):153-66.
- [23] Kitamura Y, Mizoguchi R. Organizing Knowledge about Functional Decomposition. In Proc. of the 14th International Conference on Engineering Design (ICED 03) 2003.
- [24] Kitamura Y, Mizoguchi R. Ontology-based systematization of functional knowledge. Journal of Engineering Design 2004;15(4): 327-51.
- [25] Gruber TR. A translation approach to portable ontologies. Knowledge Acquisition 1993;5(2):199-220.
- [26] Liu Z. Integrating Two Ontology for Electronics. In Recent Advances in Qualitative Physics, MIT Press; 1992, p. 153-68.
- [27] Gruber TR, Tenenbaum JM, Weber JC. Toward a Knowledge Medium for Collaborative Product Development. In Proc. of Artificial Intelligence in Design '92 1992;413-32.
- [28] Cutkosky MR, et al. PACT: An Experiment in Integrating Concurrent Engineering Systems. Computer 1993;January:28-37.
- [29] Sekiya T, Tsumaya A, Tomiyama T. Classification of Knowledge for Generating Engineering Models - A case study of model generation in finite element analysis -. In; Finger S, Tomiyama T, Mäntylä, editors. Knowledge Intensive Computer Aided Design, Boston: Kluwer Academic Publishers; 1999, p. 73-90.
- [30] De Kleer J, Brown JS. A Qualitative Physics based on Confluences. Artificial Intelligence 1984;24:7-83.
- [31] Gruber T, Olsen G. Theory Component-Assemblies, Ontology Server. <http://www-ksl.stanford.edu>. 1994.
- [32] Borst P, Akkermans H, Top J. Engineering Ontologies. Int'l Journal of Human-Computer Studies 1997;46(2/3):365-406.
- [33] Kumar AN, Upadhyaya SJ. Component-Ontological Representation of Function for Reasoning about Devices. Artificial Intelligence in Engineering 1998;12:399-415.
- [34] Koji Y, Kitamura Y, Mizoguchi R. Towards modeling design rationale of supplementary functions in conceptual design. In Proc. of Fifth International Symposium on Tools and Methods of Competitive Engineering 2003; 117-30.
- [35] van der Vegte WF, 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. To appear.
- [36] Sasajima M, Kitamura Y, Ikeda M, Mizoguchi R. FBRL: A Function and Behavior Representation Language. Proc. of IJCAI-95 1995;1830-6.
- [37] Mizoguchi R, Kitamura Y. Foundation of Knowledge Systematization: Role of Ontological Engineering. In; Rajkumar Roy, editors. Industrial Knowledge Management - A Micro Level Approach, Springer-Verlag, 2000; 17-36.
- [38] Chandrasekaran B. Design Problem Solving: A Task Analysis. AI Magazine 1990;11(4):59-71.

- [39] Forbus KD. Qualitative Process Theory. *Artificial Intelligence* 1984;24:85-168.
- [40] Salustri FA. Ontological Commitments in Knowledge-based Design Software: A Progress Report. In Proc. of the Third IFIP Working Group 5.2 Workshop on Knowledge Intensive CAD 1998;31-51.
- [41] Horváth I, Vergeest JSM, Kuczogi G. Development and Application of Design Concept Ontologies for Contextual Conceptualization. In Proc. of 1998 ASME Design Engineering Technical Conferences DETC, CD-ROM: DETC98/CIE-5701, ASME, New York. 1998.
- [42] Davies J, Fensel D, van Harmelen F, editors. *Towards the Semantic Web - Ontology-driven Knowledge Management*. John Wiley & Sons; 2003.
- [43] Rajan JR, Stone RB, Wood KL. Functional Modeling of Control Systems. In Proc. of ICED 03 2003.
- [44] Wilhelms S. A Conceptual Design Support System using Principle Solution Elements. Proc. of ICED 03 2003.
- [45] Tejima N. et al., editors. Selection of functional terms and categorization, Report 49, Soc. of Japanese Value Engineering (In Japanese), 1981.
- [46] Hirtz J, Stone RB, McAdams DA, Szykman S, Wood KL. A Functional Basis for Engineering Design: Reconciling and Evolving Previous Efforts. *Research in Engineering Design* 2002;13:65-82.
- [47] Rosenberg RC, Karnopp DC. *Introduction to Physical System Dynamics*. McGraw-Hill; 1983.
- [48] Gero JS, Kannengiesser U. The Situated Function-Behaviour-Structure Framework. In Proc. of *Artificial Intelligence in Design '02* 2002;89-104.
- [49] Sushkov VV, Mars NJI, Wognum PM. Introduction to TIPS: a Theory for Creative Design. *Artificial Intelligence in Engineering* 1995;9.
- [50] Steven K, Peder F, Ishii K. Advanced Failure Modes and Effects Analysis of Complex Processes. Proceedings of the ASME Design for Manufacturing Conference 1999.
- [51] Hata T, Kobayashi N, Kimura F, Suzuki H. Representation of Functional Relations among Parts and Its Application to Product Failure Reasoning. Proceedings of CIRP International Seminar on Design 2000.
- [52] De Kleer J, Williams BC. Diagnosing Multiple Faults. *Artificial Intelligence* 1987;32:97-130.
- [53] Larsson JE. Diagnosis based on Explicit Means-ends Models. *Artificial intelligence* 1996;80:29-93.
- [54] Davis R. Diagnostic reasoning based on structure and behavior. *Artificial Intelligence* 1984;24:347-410.
- [55] Böttcher C. No fault in structure? - how to diagnose hidden interactions. In Proceedings of IJCAI-95 1995; 1728-33.
- [56] Kitamura Y, Mizoguchi R. An Ontological Analysis of Fault Process and Category of Faults. In Proceedings of Tenth International Workshop on Principles of Diagnosis (DX-99) 1999;118-128.