

# **An Ontology-Based Framework for Systematization of Functional Knowledge and Its Use for Supporting Designers**

Yoshinobu Kitamura and Riichiro Mizoguchi

AI-TR-2002-1

Received: April 18, 2002

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

## **Abstract**

It has been recognized that design knowledge is scattered around technology and target domains. One of the reasons is that different frameworks (viewpoints) for conceptualization of design knowledge are used when people try to describe knowledge in different domains. Another one is that several key functional concepts are left undefined or even unidentified. Aiming at systematization of functional knowledge for synthesis, we discuss ontologies that guide conceptualization of artifacts from the functional point of view. We propose an extended device ontology and a functional concept ontology built on top of the device ontology. The utilization of the systematized functional knowledge in several application systems is also discussed together with its advantages.

**Keywords:** Functional representation, Functional reasoning, Ontology

## **Corresponding author:**

Yoshinobu Kitamura

The Institute of Scientific and Industrial Research, Osaka University,  
8-1, Mihogaoka, Ibaraki, Osaka, 567-0047, Japan

Phone: +81-6-6879-8416, Fax: +81-6-6879-2123

E-mail: kita@ei.sanken.osaka-u.ac.jp

# **An ontology-based framework for systematization of functional knowledge and its use for supporting designers**

YOSHINOBU KITAMURA AND RIICHIRO MIZOGUCHI

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

It has been recognized that design knowledge is scattered around technology and target domains. One of the reasons is that different frameworks (viewpoints) for conceptualization of design knowledge are used when people try to describe knowledge in different domains. The other one is that several key functional concepts are left undefined or even unidentified. Aiming at systematization of functional knowledge for synthesis, we discuss ontologies that guide conceptualization of artifacts from the functional point of view. We propose an extended device ontology and a functional concept ontology built on top of the device ontology. The utilization of the systematized functional knowledge in several application systems is also discussed together with its advantages.

## **1. Introduction**

Design is a creative activity that translates a requirement specification at the functional level into a set of attribute values of concrete things. Although advancement of computer and AI technologies has enabled easy access to information related to structure and/or shape of artifacts, design know-how used in the conceptual design phase is left implicit because each designer possesses it. As discussed in the literature on knowledge management, such subjective, and hence implicit knowledge needs to be made explicit if it is to be shared within a community. The same applies to design community and it is expected that design knowledge sharing will improve the design process drastically. In order to make it happen, however, we need to resolve some major problems. One of them is to devise a framework for capturing and describing conceptual design knowledge that has never been shared within any community. Such a framework should be general enough to be shared by people in different communities and should enable consistent description of such knowledge in a computer interpretable form. However, it is a challenge to make it possible to describe subjective design knowledge in a general and sharable form. In fact, we see many examples that fail to organize knowledge in such a way.

The main goal of our research is to design a sophisticated framework for systematization of functional design knowledge through ontological engineering (Mizoguchi & Ikeda 1997). Explication of conceptual structure behind the design knowledge contributes to interoperation between knowledge and to sharing/reuse of knowledge by providing a firm basis. Among various kinds of design knowledge, we concentrate on functional knowledge. The next section overviews the knowledge engineering that has developed into ontological engineering and describes the scope of our enterprise. Section 3 describes the skeletal plan of our ontology building for knowledge systematization. Section 4 proposes an extended device ontology that plays a crucial role in our framework. On the basis of the device ontology proposed, functional ontologies are discussed in detail in section 5. In order to demonstrate the utility of the knowledge systematization, an overview of application systems such as one for understanding functional structure of a given artifact are described in section 6. Functional knowledge description and a designer's support system that suggests some appropriate functional decomposition knowledge is described in detail in section 7. Section 8 discusses related work followed by concluding remarks.

## **2. Ontological Engineering and Knowledge Systematization**

### **2.1 From knowledge base to ontological engineering**

Expert systems that deal with real-world knowledge have already spread in every industry. Although many systems have been built for design support, few are successful. One of the problems of knowledge bases in expert systems is that they are ad-hoc and specific to a given task, and hence they are hardly reusable and far from knowledge systematization.

The recent advancement of knowledge modeling and ontological engineering shows a new direction of knowledge base technology that is summarized as follows:

Table 1 Current state of the art of design knowledge systematization.

	Knowledge about design objects	Knowledge about design process		
		Logical aspects	Mathematical aspects	System aspects
Domain dependent	Conventional domain-oriented engineering	NA	NA	Ad hoc system
Domain independent	<b>None1</b>	Tomiyaama's project	General Design Theory	<b>None2</b>

- Advancement of knowledge engineering provides us with an enabling technology for building sharable knowledge bases on which we can accumulate knowledge.
- Requirements to the knowledge base technology have changed from a powerful problem solving system for a specific task to collaborative systems that help us solve problems by managing and providing standardized and useful knowledge of wide spectrum of domains.

This trend suggests that *ontological engineering* would be the next enabling technology. An ontology, which is a system of knowledge of high universality, that is, a system of background knowledge of any knowledge base, explicates the conceptualization of the target world and provides us with a solid foundation on which we can build sharable knowledge bases for wider usability than that of a conventional knowledge base. Knowledge engineering has thus developed into ontological engineering. Our enterprise is to build an innovative framework for knowledge systematization for wider use of functional design knowledge on the basis of ontological engineering.

## 2.2 Policy of systematization

Table 1 represents our understanding of the current state of the art of design knowledge systematization. Basically, the conventional research on design knowledge is highly domain-dependent and has investigated such knowledge by analytical and quantitative methods. Knowledge base construction has been ad-hoc as is seen in the upper row of Table 1. Yoshikawa (1981) initiated the research on General Design Theory (GDT) to overcome the difficulties caused by the domain-dependence of the research activities. GDT is mainly concerned with the static nature of design in terms of mathematics, and hence it does not cover the dynamic aspects of design, that is, design process. Tomiyama's project (Tomiyama 2000) has been investigating the research on design process modeling using logic and artificial intelligence and has come up with a deeper understanding of design process, that is, "design process is an abduction". Table 1 shows there are two major issues left untouched. In our research, we have been mainly attacking the issue depicted as **None1**. Although it is mainly concerned with *knowledge* rather than *process*, our research shares the philosophy with Yoshikawa and Tomiyama in the sense that *investigating synthesis from a domain-independent view to reveal the inherent nature of synthesis*.

By building a framework for knowledge systematization using ontological engineering, we mean identifying a set of backbone concepts with machine understandable description in terms of which we can describe and organize design knowledge for use across multiple domains. The system of concepts is organized as layered ontologies as is seen in the next section.

## 3. Functional Ontology and Knowledge Systematization

No one would disagree that the concept of function should be treated as a first class category in design knowledge organization. That is, function is an important member of a top-level ontology of design world. One of the key claims of our knowledge systematization is that the concept of function should be defined independently of an object that can possess it and of its realization method. Contrary to the possible strong disagreement from the people working in the mechanical design, the claim has a strong justification that the concept of a function originally came from the user requirements which is totally object- and behavior-independent, since common people have no knowledge about how to realize their requirements and are interested only in satisfaction of their requirement by a device built. Another justification is reusability of functional knowledge. If functions are defined depending on objects and their realization, few functions are reused in and transferred to different domains. As is well understood, innovative design can be facilitated by flexible application of knowledge or ideas across domains.

### 3.1 Functional representation

Functional representation has been extensively investigated to date (Keuneke 1991; Chandrasekaran, Goel & Iwasaki 1993; Chittaro, Guida, Tasso & Toppano 1993; Lind 1994; Sasajima, Kitamura, Ikeda & Mizoguchi 1995; Umeda, Ishii, Yoshioka, Shimomura & Tomiyama 1996) and a lot of functional representation languages are proposed with sample descriptions of functions of devices. However, because it is not well understood how to organize functional knowledge in what principle in terms of what concepts, most of the representation are ad-hoc and lack generality and consistency, which prevents knowledge from being shared. One of the major causes of the lack of consistency is the difference between the ways of how to capture the target world. For example, let us take the function of a super heater of a power plant, *to heat steam* and that of cam of cam&shaft pair, *to push up the shaft*. The former is concerned with something that comes in and goes out of the device but the latter with the other device that cannot be either input or output of the device. This clearly shows the fact that there is a difference in how to view a function according to the domain. The difference will be one of the cause of inconsistency in functional representation and non-interoperability of the knowledge when functional knowledge from different domains is put into a knowledge base.

The above observation shows that we need a framework which provides us with a viewpoint to guide the modeling process of artifacts as well as primitive concepts in terms of which functional knowledge is described in order to come up with consistent and sharable knowledge. However, conventional research of function is not mature enough to propose such a framework and only a few functional concepts are identified to date (Pahl & Beitz 1988; Chittaro et al. 1993; Lind 1994). Although value engineering (Miles 1961; Tejima et al. 1981) has a long history of research on functional terms and has come up with a rich set of functional vocabulary, they are for human consumption and no operational definition is made.

### 3.2 Hierarchy of functional knowledge and ontology

Figure 1 shows a hierarchy of functional knowledge built on top of fundamental ontologies. The lower layer knowledge is in, the more basic. Basically, knowledge in a certain layer is described in terms of the concepts in the lower layer. Top-level ontology defines and provides very basic concepts such as time, state, process and so on. Causal ontology specifies actions and causality against teleology. Physical world ontology specifies 3D space and entity to give axiomatic physical world with a state-based modeling reflecting a special world of design in which an entity(artifact) is created from nothing. These two ontology contributes to “Symbol grounding” of higher-level concepts, that is, functional concepts. Although these two topics are very important, they are omitted in this paper due to the space limitation. On top of these three, physical process ontology is introduced to specify physical(natural) processes. A very primitive definition of *process* is done in the top-level ontology to cover any sequence of events in terms of state and its change. The term “Process” is confusing, since it has three different meanings; sequence/course/stage, something found in the phrase “Burning process” and processing. The first is one defined in the top-level ontology, the second is covered by the physical process ontology introduced here and the last is covered by device ontology discussed in section 4. These five ontologies(Top-level, causality, physical world, physical process and device ontologies) collectively work as a substrate on which we can build consistent knowledge in layers.

Functional concept ontology specifies functional concepts as an instance of *function* defined in device ontology. The definitions are scarcely depends on the device, the domain or the way of its implementation so that they are very general and usable in a wide range of areas. Theories and principles of physics and abstract part library also belong to this class of knowledge called *general concept layer*.

Way of functional achievement is knowledge about *how*(in what way) a function is achieved, whereas the functional concept is about *what* the function is going to achieve. Although functional achievement way knowledge looks similar to functional decomposition like that discussed in (Pahl & Beitz 1988), the former is

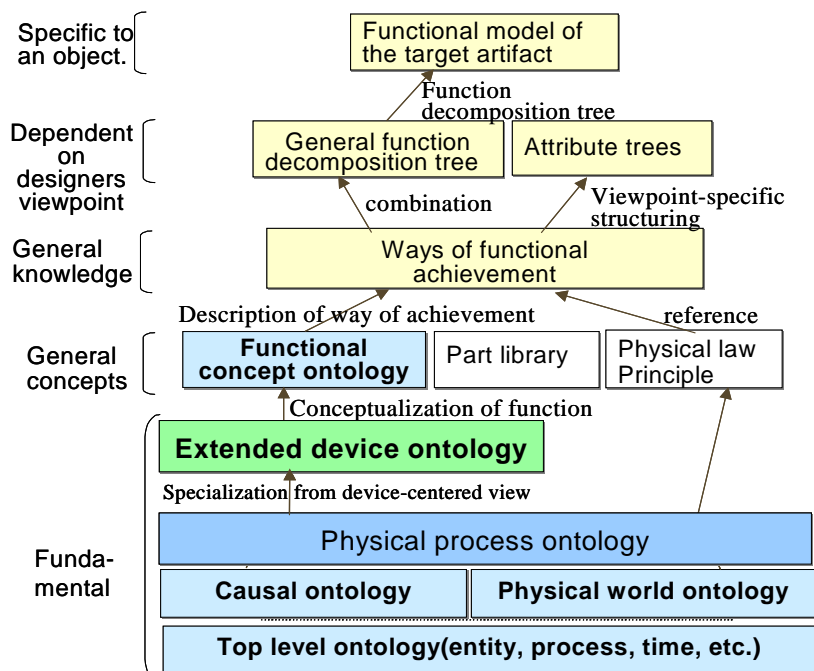


Figure. 1 Hierarchy of ontology and knowledge.

much richer than the latter in that it consists of four kinds of hierarchies of different roles and principles(See section 7). The inherent structure of such knowledge is organized in an *is-a* hierarchy from which the other three structures are derived according to the requirement. The *is-a* structure is carefully designed identifying inherent property of each *way* to make it sharable and applicable across domains. One of the key issues in knowledge organization is clear and consistent differentiation of *is-a relation* from other relations such as *part-of*, *is-achieved-by*, etc. keeping what is the inherent property of the target thing in mind. The next section is devoted to the intensive discussion on device ontology that is the key topic of this paper. All the concepts introduced and discussed in the rest of the paper are based on the extended device ontology we propose.

## 4. Device Ontology

### 4.1 What is device ontology?

Concerning modelling of artifacts, there exist two major viewpoints: Device-centered and Process-centered views. The device ontology, e.g., one proposed by de Kleer and Brown (1984), specifies the device-centered view of artifacts. Device-centered view regards any artifact as composition of devices which process input to produce output which is what the users need. Process-centered view applied to an artifact, e.g., one proposed by Forbus (1984), concentrates on phenomena occurring in each artifact(device) to obtain the output result with paying little attention to the devices existing there. Device ontology imposes a frame or viewpoint on an event to introduce an engineering perspective. That is, it introduces the concepts of a black box equipped with input and output ports. Although physical process ontology, which specifies the process-centered view, is more fundamental than device ontology, there are some cases where process ontology is directly employed to model real world events/phenomena instead of device ontology. Typical cases are found in modelling chemical processes for which device ontology is not appropriate

The major difference between the two is that while device ontology has an **Agent**, which is considered as something that plays the main actor role in obtaining the output, process ontology does not have such an **Agent** but has **participants**, which only participate in the phenomena being occurring. Needless to say, such an **Agent** coincides with a device in device ontology.

Device ontology specifies the roles played by the elements that collectively constitute a device. The concept of role is a hot topic in ontological engineering because an object plays different roles in different situations, and the fact has been a major source of failure in conceptualization of the world. For example, a man plays many roles such as husband, father, son in his family. These roles are defined in the family context, and hence they are specified by family ontology. Thus, device ontology can be said as a role specification system for the elements we recognize in a device in general.

### 4.2 Why device ontology

In spite of its less basic characteristics, device ontology has been dominant to date in modelling artifacts for many reasons as follows:

1. The idea of device ontology is straightforward because most artifacts are device in some sense.
2. Every artifact is easily and nicely modelled as a part-whole hierarchy of devices because it can be considered as being composed of sub-devices recursively.
3. The concept of a function has to be attributed to an agent(a device), which is viewed as a main actor to achieve the function.
4. Device-centered view saves a load of reasoning about design; it hides internal details of devices.
5. One can design almost all artifacts in the device ontology world by configuring devices, except cases where one needs innovative devices based on new combination of phenomena or new phenomena themselves.

### 4.3 Limitation of device ontology and its extension

A naïve idea of device ontology was born in system engineering. It is composed of components and connection between them and has been extensively used in many engineering areas as well as in design community as is discussed in (Pahl & Beitz 1988). However, it has no criterion of which role should be played by which component and the assumptions behind the ontology is implicit, and therefore modelling of artifacts based on that ontology can be ad-hoc. Even worse, it is hard to compare it against other ontologies and its limitations are not clear.

De Kleer and Brown (1984) introduced an idea of conduit into the naïve device ontology so that it is easy to distinguish device from conduit. His ontology still leaves the identification of objects unclear. Umeda et al. (1996) points out the limitation of functional decomposition proposed by Pahl and Beitz (1988) based on device ontology and Mortensen (1999) reports on a negative observation on the applicability of device ontology to mechanical

elements. Our major objectives include to understand in-depth the capability and limitations of device ontology together with its extension if necessary in order to have a common foundation for building artifact models usable across domains.

It is important to note, however, device ontology is not almighty. Device ontology requires description of each device in advance, therefore creative results are expected only by combination of device models available rather than by creating a new device. This means that device ontology never promises invention of a new element of an innovative shape that is important in mechanical design. However, many of the innovative design have been done a new combination of known devices or components, especially in domains except mechanics (Sushkov, Mars, Wognum, 1995). Another typical research topic in mechanical domain is to predict behavior from a new shape of element, which is also out of device ontology's responsibility. In spite of these limitations, we take device ontology as our guiding principle, since we aim at coming up with a framework covering wider range of areas.

Essentially, device ontology views any device as a black box. Of course, in the device ontology world, one can go into smaller grained world in which the parent device is seen as configuration of several sub-devices of smaller grain size. The smaller devices themselves are still devices whose internal behavior is also hidden. Process ontology is different. It is useful for explaining why a device works, since it directly explains what phenomena are occurring in the device to achieve its main function. Furthermore, process ontology is better than device ontology at modelling, say, chemical plants where chemical reactions have to be modelled as phenomena.

Our claim is not that device ontology enables all kinds of description of all kinds of artifacts but that we should appreciate the potentials of device ontology and we do our best to extend it if possible to extend its applicability without losing its advantages.

#### 4.4 Extended device ontology

This section presents the key concepts(roles played by objects) in the extended device ontology. The discrimination between behavior and function is discussed in section 5. We exclude static behavior such as *to support* by concentrating only on dynamic behaviors.

##### 4.4.1 Device and object

Things that exist in the device ontology world are categorized into two categories: *Device* and *Object*. A *device* has input and output ports through which it is connected to another device(precisely speaking, not a device but conduit which is explained later in detail). A *device* consists of other devices of smaller grain size and usually is organized in a whole-part hierarchy of sub-devices. An *object*<sup>1</sup> is something that can be considered as that it goes through a device from the input port to the output port during which it is processed by the device. Examples of an *object* include substance like fluid, energy like heat, motion, force, information, etc. An *object* has attributes whose values change over time.

A *device* is something that operates on an *object* that goes through the device. The state change of an *object* is realized by the difference between the states of the *object* at the input port and that at the output port. What happens inside the device is invisible at that grain size. Although it seems to be visible when going down into the level of smaller grain size, what is seen there is different from what would be seen when one looks into a device at the same grain size, which is discussed in 4.4.3.

##### 4.4.2 Conduit and medium

A *conduit* is defined as a special type of *device* that can be considered as it transmits an *object* to output port without any change in an ideal situation. Examples include a pipe, a shaft, etc. Although the pressure of fluid that passes through a pipe is decreased in reality, there is no problem if we neglect the loss in an ideal situation. So, a pipe is a conduit as a special type of device. We model a pipe in the same way as we do device. The difference between the two is cognitive. In our extended device ontology, a conduit does not have to be substantial but can be virtual. In the case of mechanical element level modelling, we introduce virtual conduit between elements as will be discussed in 4.5.

A *medium* is something that holds an *object* and enables it to flow among devices. For example, steam can play the role of a medium because it can hold heat energy. In some domain, conduit can play the role of medium. For example, while a shaft is a conduit for force and motion, at the same time, it plays the role of medium for them. A medium may flow to carry an object like fluid flows to carry heat energy, but it does not have to. A medium may

---

<sup>1</sup> The label of this concept "object" represents neither "entity", "substance" nor "purpose" but "target" of activity of a device.

stay at the same position and transmit an object by passing it to the connecting medium like a shaft does.

#### 4.4.3 Behavior

We identified four kinds of definitions of *Behavior*. Figure 2 illustrates simplified situations for behavior definition. **B0 behavior** is defined as the change of an attribute value of an *object* at the same location over time. Typical example is increase of the temperature of fluid at some observation point over time. Note that what is observed is a different thing at any time. This is exactly same as the observation of a real phenomenon and coincides with what numerical simulation obtains.

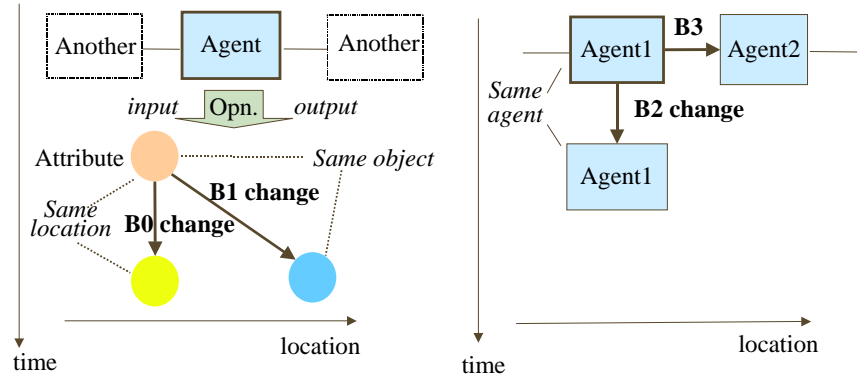


Figure 2 Four different definitions of behavior.

**B1 behavior** is defined as the change of an attribute value of an *object* from that at the input port of a device to that at the output of the device. For example, the increase of the temperature of steam occurred during it goes through a super-heater is B1 behavior. The key difference between B0 and B1 is that while B0 behavior concentrates on the location of the observation rather than identity of the observed *object*, B1 behavior on the identity of the observed *object* rather than the location.

**B2 behavior** is defined as the change of something inside of a device rather than input/output ports. The “something” could be *motion* of a part of the device or inner state of the device. For example, “rotation of fins in a fan” is an inner behavior of “to fan”, “a shaft is twisted” is an inner behavior of “to transmit torque”, etc. This behavior is based on an answer to the question such as “what motion is the device performing?” and is not the behavior of the device of smaller grain size but that identified by peeping into the device with a violation of the “black box principle” of device ontology.

**B3 behavior** is defined as any behavior to another *device*. The important aspect here is *B0* and *B1* behaviors are concerned with *objects* rather than *devices*.

All the definitions above share that behavior is a conceptualization of the change of attribute values in the spatio-temporal space over time. The differences come from the way of treatment of the location in the spatio-temporal space and the target of the operation to be interpreted as behavior. Another definition of behavior, which looks very similar to B1 behavior at first glance, is found in (Chandrasekaran et al. 1993) where function is defined as *B1 behavior* and the behavior corresponding to the function is defined as series of *B1 behavior* of sub-devices of smaller-grain size. That is, in his definition, the difference between function and behavior is relative to the grain size, which is different from our principle that function is a teleological interpretation of behavior.

In our extended device ontology, we adopt **B1 behavior** as the proper definition of behavior of a device in the device-centered world. Before explaining the justification of adopting B1 behavior, let us defend our definition of **B1 Behavior** from a possible critic that it seems not compliant to what a normal simulation does. The counter argument would look like as follows:

*In the world of numerical simulation, which is the typical way of simulation, by simulation, we mean trace of the change of a parameter value over time. In other words, the behavior is an interpretation of change of each parameter value. It is straightforward and convincing, since it is compatible to the basis idea of how things behave which is not explained by B1 behavior.*

This view is parallel to the naïve view of behavior like the ideas of *B0* and *B2* behaviors based on “motion” such as walking and running. It is true that the above idea of numerical simulation is basic because it depends on neither device nor process ontologies. There is, however, another kind of simulation, that is, *causal simulation* in qualitative reasoning. If we interpret it in the device ontology world, causal simulation is different from the numerical simulation in that it tries to simulate the change of an object between that when it exists at the inlet of the device and that when it exists at the outlet of the device, that is, causality of the change made by the device, which is not captured by numerical simulation and is what AI research needs about behavior of a system from device-centered point of view.

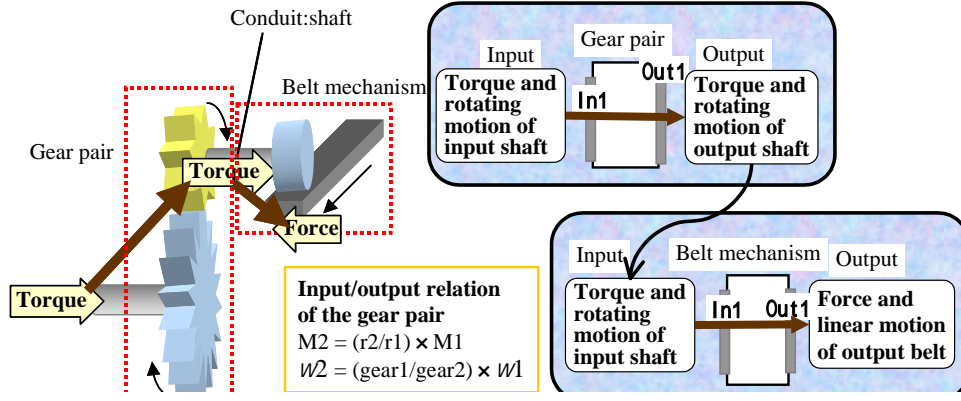


Figure.3a A model of a pair of gears at the mechanism level

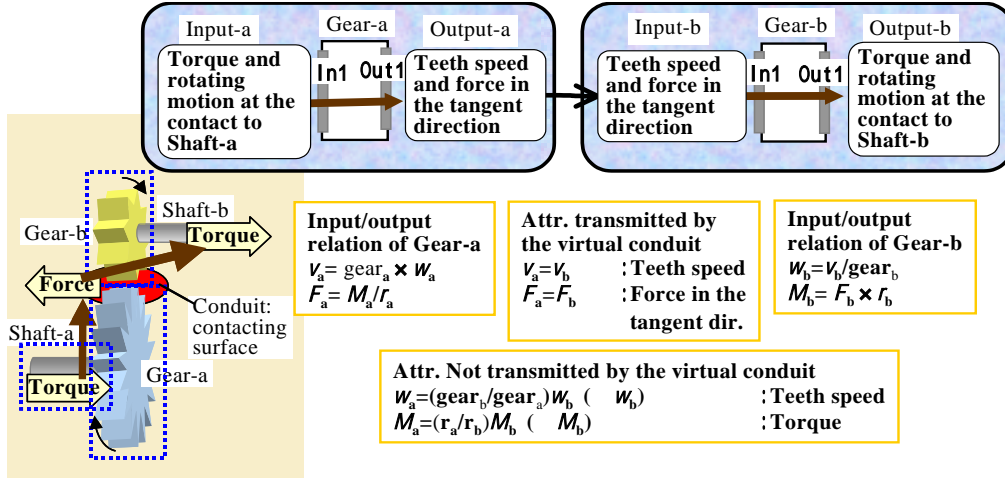


Figure 3b A model of a pair of gears at the mechanical element level.

In design community, what is necessary are the ideas of function and behavior as well as structure. In order to relate the idea of function to that of behavior, both should be based on the same ontology, or at least on the ontologies mutually compatible. As discussed above, functional concepts inherently need device ontology because we always associate function with a device. This suggests us that behavior should be defined in terms of device ontology, that is, **B1 behavior** that is consistent with not the idea of numerical simulation but that of causal simulation. In the rest of this article, we use the term “behavior” instead of **B1 behavior** for simplicity.

#### 4.5 Modelling of mechanical systems based on the extended ontology

In the extended device ontology, we view motion and force in mechanical systems as an *object*. That is, a mechanism as a device is considered as a thing that changes attributes(direction, amount, etc.) of motion and force.

There are two levels of grain sizes in mechanical systems: mechanism level and mechanical element level. By a mechanical element, we mean a gear, a shaft and so on and by a mechanism, we mean a complex of elements like a gear pair. Identification of a mechanism is done by identifying a conduit and by considering it as the boundary between mechanisms. A conduit at the mechanism level is a shaft or a wire, since they just transmit force or motion without change from one end to the other end. Let us take an example shown in Figure 3a. The gear pair is modelled as a device that accepts torque and angle velocity as input through a shaft as a conduit and outputs them with different values determined according to the ratio of gear numbers. The output is put on the shaft and transmitted to the belt mechanism that changes the rotation motion to horizontal motion. These two mechanisms constitute a larger mechanism.

Treatment of elements is as follows: A conduit at the element level is virtual and is defined as conceptualization of the mechanical pair(of elements), that is, the contact point/line/face that locates at the boundary between the connecting elements. A wheel is modelled as a device that has a line-contact conduit that can transmit only tangent velocity and force at the surface and has fixed(embedded) connection to a shaft around its central part that transmits everything. A gear in a gear pair is modelled in the similar way as a wheel. Let us take an example shown in Figure 3b. A gear is modelled as a device that accepts number of rotations and torque and outputs tooth number velocity(number of teeth per time) and tangent force that is obtained by dividing the torque by radius of



Table 2 Types of conduit.

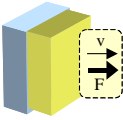
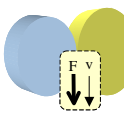
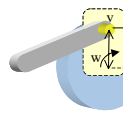
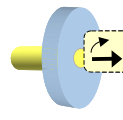
Domain	Plant	Mechanical domain: Mechanism level		Mechanical domain: Mechanical element level			
		Shaft	Wire	Surface contact	Line/point contact	Rotation contact	Fixed contact
Kind	Pipe						
Shape		Cylinder	Line	Plane	Plane-cylinder Cylinder-cylinder	Circles of the common axes	-
What is transmitted	All attributes of object	All forces and motion	Drawing force	Liner motion and forces vertical to the plane	Linear motion and force in the tangent direction of the contact	Linear and circular motion in the direction vertical to the axis	All forces and motion

Table 3 Comparison among key concepts in plant and mechanical system domains.

	Plant: Energy	Plant: Entity	Mechanical: Mechanism level	Mechanical Mech. element level
<i>Device</i>	Boiler, turbine, etc.	Boiler, Distiller, etc.	Mechanism (Gear pair, Cam&shaft, etc.)	Mech. Element (gear, shaft, etc.)
<i>Conduit</i>	Pipe	Pipe, Belt conveyer	Shaft and wire	Contact (Surface, line, point)
<i>Object</i>	Heat energy	Fluid, stuff, etc.	Force and motion	Force and motion
<i>Medium</i>	Fluid (water, steam, etc.)	Fluid, tool, or nothing	Shaft and wire	Contact
<i>Function</i>	generate, give, rob, cool, etc.	divide, distil, separate, process, etc.	change No. of rotation, change kind of motion, etc.	change speed, transmit force and motion, etc.

the gear. This model neglects the boundary between teeth, though it causes no problem in our goal. Note here that input torque is not transmitted by a gear or a wheel as a mechanical element, since torque can be changed according to the ratio of the number of teeth of the connecting gears or to the ratio of the radiuses of the wheels.

Another important issue here is what the virtual conduit is. Table 2 summarizes the concept of a conduit in domains such as plant systems and mechanical systems. The inherent property of a conduit in the device ontology is that it transmits all attributes that medium holds. It is true for pipe in the plant domain and a shaft in the mechanical domain. But, a wire transmits only pulling force and the virtual conduit introduced in the element level of mechanical domain is not the case. It transmits only limited attributes depending on the pair of the elements. This is the key extension of our device ontology. Although table 2 shows only a portion of kinematic pairs as a conduit, it is easy to define other conduit for rest of the mechanical pairs.

#### 4.6 Applicability of the extended device ontology to different domains

The contributions of the extended device ontology are two fold: it provides us with (1) a machine understandable framework for modelling artifacts in different domains with a consistent viewpoint and (2) appropriate vocabulary in terms of which we can describe differences between models in different domains. Table 3 shows comparison among models in plant domain and mechanical system domain. The differences are summarised as follows: In the mechanical system domain,

- (1) A *conduit* is degenerated to *medium*. In other words, the *conduit role* and *medium role* are played by a single thing at the same time.
- (2) *Medium* does not flow through a *device*, but it allows *objects* to flow by transmitting it through their connection.
- (3) Force and motion are *objects* processed by a *device*.
- (4) *Conduit* at the mechanical element level is virtual.
- (5) What is transmitted by a *conduit* is limited depending on the types of the kinematic pair.
- (6) *Conduit* is not unique but of variety.

In spite of these differences, the extended ontology captures essential properties of models in the two domains with explication of their differences. These results shown in Tables 2 and 3 are based on our research experiences on Plant, production process (Mizoguchi & Kitamura 2000) and mechanical system domains for years.

#### 4.7 Inter-device operation ontology

Following the extended device ontology, one is not allowed to say “the gear rotates the connecting gear” or “the cam push up the rod”, etc., since it is based not on *B1 behavior* but on *B3 behavior*. We could design another ontology based on *B3 behavior* that would be called Inter-device operation ontology. Such an ontology allows you to say “device A *Verb* device B”. Needless to say, this ontology is not good in the sense of compositionality which is the key advantage of device ontology, since every behavior is dependent on neighboring devices. The device ontology can treat interaction among components using compositional component models and connections between them.

### 5. Functional Ontology

We now have obtained a framework for building of a functional model of an artifact. The next things we need are well-defined fundamental concepts in terms of which we can describe functional knowledge. In this section, we introduce several categories such as *base function*, *meta-function*, *way of functional achievement* and *method of functional achievement* together with functional concept ontology.

#### 5.1 What a function is

Typical examples of function are seen in the phrases of “this machine has speaking and speech recognition functions” or “he temporarily lost walking function”. This is one of the sources of confusions in understanding what a function is. This view tells us every action itself can be a function, because function is interpreted as capability of performing specified behavior. In design research, however, what we need is not such a view of a function but one which contributes to distinction between behavior and function because functional concepts are usually found in the requirement specification and behavior is something to realize it.

It is very important to note that by the above phrase “behavior is something to realize it(function)”, we do not mean we go inside of a device to explain the mechanism of how it realizes the function but mean a different way of interpretation of the same event at the same level of grain size as is seen in our definitions of behavior and function. Going into finer-grained explanation (understanding) is done by “functional decomposition” or “behavioral decomposition” when necessary. Note also that functional decomposition still produces another set of functions based on device ontology rather than explaining internal behavior of the device.

We define a **Function** of a device as “teleological interpretation of *B1 behavior* under a given goal” (Sasajima et al. 1995). This tells us a function of a device is determined context-dependently, though *B1 behavior* is constant independently of the context. Considering that, in most cases of design, the context of a device is determined by the goal to be achieved by the device, function of a device is determined goal- or purpose-dependently. This reflects the reality that definition of function tends to be context-dependent and hence, in many cases, functional knowledge about design is hardly reusable. The major goals of our research include to give a framework for organizing functional concepts in a reusable manner and to define them operationally as an essential step towards knowledge systematization. By operationally, we mean computers can make use of functional knowledge in the reasoning tasks of the functional modelling, understanding of the functional structure of a device and revising it. What we have to do for these goals are as follows:

1. To define functional concepts independently of its realization so as to make the definition reusable.
2. To devise a functional modelling method to enable a modeller to relate such reusable functional definitions to specific application problems, that is, to get functional concepts **grounded** onto the behavior and hence structure.
3. To formulate a functional decomposition scheme to obtain efficient functional knowledge for design.
4. To identify categories of functional concepts for systematization of functional knowledge.
5. To provide rich vocabulary for reasoning in the functional space.

The following is an overview of our work on ontologies of function.

#### 5.2 Structure-behavior-function hierarchy

Figure 4 shows Structure-Behavior-Function hierarchy. It looks similar to Figure 1 but is different from Figure 1 which is an abstraction hierarchy of concepts related to function of artifacts. By structure, we mean topological relations among components(devices). **Structure** of a device constitutes a hierarchical structure according to the grain size that is shown in the lowest plain in Figure 4. By **behavior**, we mean *B1 behavior*. What is obtained by

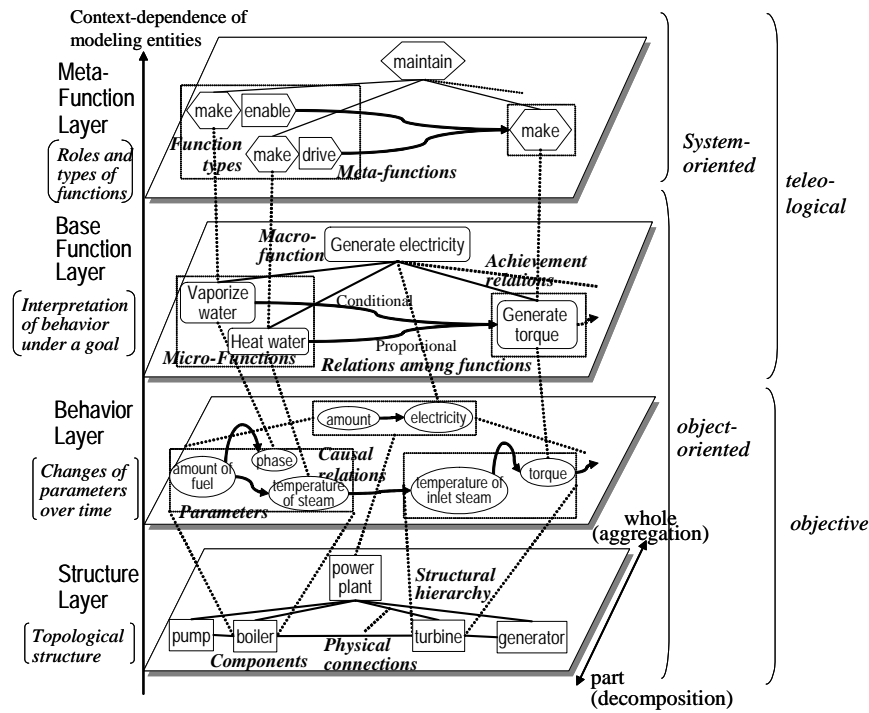


Figure 4. Hierarchy of target objects (Power plant).

teleological interpretation of *BI* behavior under a given goal is called (**Base**) *function*. The term “base” is used to discriminate it from meta-function introduced later.

A function is achieved by performing(achieving) a series of sub-functions which is called a **method of functional achievement**. On the other hand, a conceptualization of the principle or intended phenomena or structure that gives justification of how the method achieves the function is called **way of functional achievement** that is considered as reference to the essential property of structure and behavior that achieve the function. The functional decomposition based on the way of functional achievement is compliant with the observations found in the research on design processes (Takeda et al. 1994) in which they say functional decomposition is not done solely in the functional space but done by going back and forth between the functional, behavioral and structural spaces during which some portions of the artifact are determined in each of the spaces simultaneously.

Note that whole-part hierarchies in the different layers do not always correspond to each other. Although the typical functional structure is one analog to the structural hierarchy, it could have many other different hierarchies according to the viewpoint to organize functional components.

**Meta-function** is a conceptualization of type of a base function and inter-dependency between them. While a *base function* is concerned with the change of *objects* in the domain, meta-function is concerned with *base functions*. **Meta-function** as inter-dependency between base functions is defined as teleological interpretation of causal relation between *base functions*.

### 5.3 Function behavior representation language: FBRL

FBRL: Function Behavior Representation Language (Sasajima et al. 1995) is designed intended to ground functional concepts onto behavior and structure of a device. It is a language for representing a base function based on our extended device ontology. It consists roughly of input and output ports for device connection, behavioral definition in terms of attributes of *objects* and functional toppings(FT) that enable a system to map a structural and behavioral model to a functional concept. FT is composed of four items:

- (1) *Obj-Focus* specifies *objects* to focus on
- (2) *O-focus* specifies attributes of the *object* to focus on
- (3) *P-Focus* specifies port to focus on
- (4) *Necessity* specifies the necessity of *objects*

### 5.4 Functional concept ontology

Functional concept ontology defines three kinds of functional concepts introduced in 5.2 (Kitamura & Mizoguchi 1999a, 2000). Figure 5 shows a portion of *is-a* hierarchy of those concepts. Base function consists of four kinds of functions such as function for substance, that for energy, that for information and that for force&motion. Figure 5

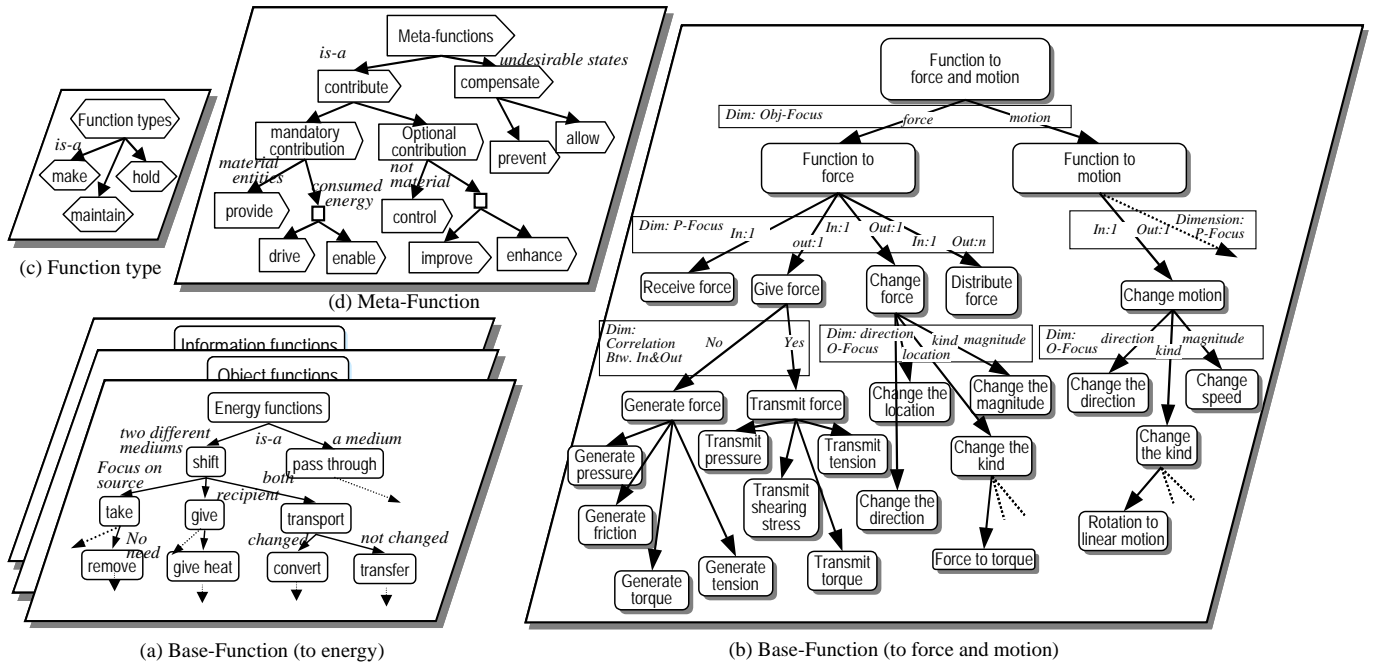


Figure 5. Functional concept ontology (portion)

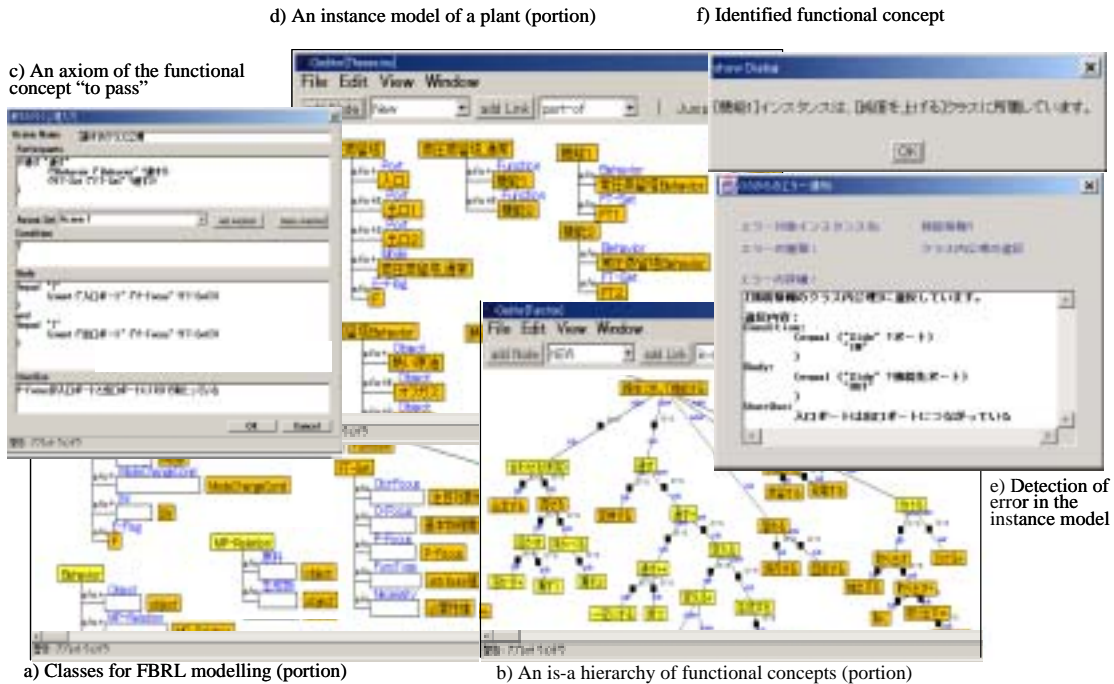


Figure 6. Snapshot of implementation of functional concept ontology

shows *is-a* hierarchies of functional concepts for energy(5a) and force&motion(5b). Each concept in the hierarchies are defined using FBRL and hence operational. For example, an energy function, *To take*, is defined as a behavioral constraint: *to shift* of energy and P-Focus on the port of energy provider. *To remove* is defined as that of *to take* with an additional FT, the energy taken is *unnecessary* as *Necessity* FT. These definitions demonstrate high independence of their implementation, while function is clearly related to structure and behavior. As is also shown in Figure 5b, functional concepts that mainly appear in mechanical system domain are defined in the same way as others. Note here that because the definitions are based on the extended device ontology, all the functional concepts(*verbs*) have *objects(force and motion)* rather than *devices(mechanisms and mechanical elements)* as their grammatical object.

The functional concept ontology is defined using Hozo, an environment for ontology development and use (Kozaki, Kitamura, Ikeda & Mizoguchi 2000). Figure 6 shows screen dump of Hozo in which (b) shows an *is-a* hierarchy of functional concepts(a shadowed rectangle represents a functional concept) and (c) shows an axiom of

the functional concept, *To pass*. Concepts used in FBRL modelling are defined in (a) and are used to build an instance model of a real-scale Crude refinery plant shown in (d). Functional concepts(terms) are identified as shown in (f) using the functional concepts defined in (b) and axiom interpretation system successfully. For details of Hozo, see (Kozaki et al. 2000).

The function types represent the types of goal achieved by the function (Keuneke 1991). Keuneke proposes some function types including “*To Prevent*” which represents to “keep a system out of an undesirable state of objects”. However, because it focuses on changes of objects associated with the component, the objective of the function is implicit, that is, another function would be affected by the state. Therefore, we redefined the function type as “*To Make*”, “*To Maintain*”, and “*To Hold*” and redefined “*To Prevent*” as a kind of a meta-function as below.

**Meta-function** (denoted by  $mf$ ) represents a role of a *base function* called an *agent function* ( $f_a$ ) for another *base function* called a *target function* ( $f_t$ ) (Kitamura & Mizoguchi 1999a, Kitamura, Sano & Mizoguchi 2000). We have defined the eight types of *meta-functions* as shown in Figure 5d (an *is-a* hierarchy). We begin definition of *meta-functions* with the condition where there is a causal relation from the focused parameter of  $f_a$  to that of  $f_t$ . If the goal of  $f_t$  is not satisfied when  $f_a$  is not achieved, the  $f_a$  is said to have a *mandatory contribution* for the  $f_t$ . Although we can intuitively say that  $f_a$  has a *To Enable meta-function* for  $f_t$  in such a case, the authors define a narrower meaning of *To Enable* by excluding the cases of *To Provide* and *To Drive* as follows.

Firstly, when a function  $f_a$  generates such an object (or energy) that will be a part of the focused entity of  $f_t$  (called *material*), the function is said to perform a *meta-function* “to provide material” for  $f_t$ . When a function  $f_a$  generates or transfers such an energy that intentionally consumed by  $f_t$  (called *driving energy*), the function is said to have the meta-function “to drive  $f_t$ ”. Lastly, *To Enable meta-function* is used for changing a necessary condition for  $f_t$  excepting the cases of *To Provide* and *To Drive*. What we mean by this weak definition is that the conditions such as the existence of the material and that of the driving energy are too obvious to be said to enable a function.

Furthermore, a function  $f_a$  having positive effects on the undesirable side effect of a function  $f_{t1}$  is said to have a meta-function “*To allow the side-effects of  $f_{t1}$* ”. On the other hand, if a serious trouble (e.g., faults) is caused in a function  $f_{t2}$  when a function  $f_a$  is not achieved, the function  $f_a$  is said to have a meta-function “*To prevent malfunction of  $f_{t2}$* ”. The details of definitions and examples are shown in (Kitamura and Mizoguchi 1999a).

Generally speaking, one cannot claim the theoretical completeness of something like functional ontology. What one can do is to provide convincing discussion with empirical studies. We already know there is another type of meta-function such as *To synchronize* in timing-related domain.

## 6. Roles and Effects of Functional Ontology

### 6.1 Roles and effects of the extended device ontology

The extended device ontology views an artifact as something that receives input, process and outputs *objects*. The *object* is something processed by the device during it goes through a device and hence it never be another device that cannot go through a device. This ontology imposes a proper viewpoint from which one can successfully model a mechanical system in a way consistent with those models of engineering artifacts produced in other domains. It is not an easy task to build models of a lot of artifacts in a consistent way. “A gear pair changes torque”, “A cam shrinks a spring” and “A cam pushes up a rod” are inconsistent with each other in the hidden computational models. While the first one is based on the extended device ontology, the latter two are based on a different ontology, say, inter-device operation ontology. The organization of knowledge including these models will lose consistency.

The extended device ontology allows us to build interoperable models and provides us with a guideline for modelling process. For example, the concept of a *conduit* helps us consistently recognize devices by taking it as the boundary between the devices. In the mechanical system domain, a shaft and a wire, which play the role of *conduit* in the mechanism level, enable us to identify each mechanism composed of mechanical elements. Models designed based on the extended device ontology has a high composability thanks to its localized description, that is, its independence of neighboring devices that are connected to each other only through attributes of an *object*. On the contrary, composability of inter-device operation ontology is low due to its high dependence on neighboring devices.

The extended device ontology provides a unified framework in which one can build compatible models in various domains including mechanical systems and common vocabulary. In fact, terms at the higher level abstraction are common. Although motion has a special attribute of direction not common to others, most of the rest can be treated by a common framework.

Two kinds of ontologies: the extended device ontology and inter-device operation ontology can successfully distinguish two different types of terms such as *to transmit(force)* and *to move(a box)* where the former is based on (extended) device ontology and the latter on inter-device operation ontology. Both are incompatible. Figure 5b represents a functional concept hierarchy in the mechanical system domain. These functional concepts are bit unfamiliar to expert in that domain. We found that most of familiar terms in mechanical system domain belong to inter-device operation ontology which is incompatible to device ontology, and hence to the model/knowledge in other domains.

## 6.2 Use of functional concept ontology

Functional concept ontology provides us with necessary and sufficient operational terms used for representing functional knowledge/model together with constraints satisfied by them. Its utility is summarized as follows:

- **Functional model description:** A functional model of a device is easily built by putting FT to the behavioral model and the model is automatically mapped to a functional concept(term)(See Figure 6f). Such a mapping can be onto multiple functional concepts and be chosen according to the necessity. Furthermore, meta-functions can identify the relation between functions (Sasajima et al. 1995, Kitamura et al. 2000).
- **Functional knowledge description:** General functional knowledge is also modelled in the same way. The well-defined and well-organized functional terms contribute to increase of the sharability and reusability. Description and use of the knowledge of way of functional achievement is discussed later.
- **Explanation generation:** Concept/terms in the functional concept ontology are used as words in explanations generated. Thanks to the operational definitions of them, the selection of words and determination of the abstraction level at which the explanation is generated can be done flexibly (Sasajima et al. 1995).
- **Specification of the inference space:** Problem solving such as design and diagnosis can be done in the functional space that is necessary for taking into account requirements represented in terms of functional concepts. The inference space is specified by the functional concepts organized in the functional concept ontology. Because the functional decomposition knowledge is represented in terms of such functional terms, automatic functional decomposition can be done which is used for functional structure improvement system (Kitamura & Mizoguchi 1999b).
- **Automatic identification of functional structure:** The ontology enables automatic identification of functional structures of a given structure and device model with the help of meta-functions (Kitamura et al. 2000).

The above types of use and possible utility of our ontologies are investigated by us. The following can be other use(explanation is omitted here):

- **Description of design rationale:**
- **Functional indexing of design cases:**

## 6.3 Functional structure understanding (Kitamura et al. 2000)

We define a functional structure understanding task as identifying functional structures of an artifact from its structural and behavioral models, where by behavioral model, we mean a behavioral model of each component of the artifact rather than a model of the whole artifact and by the structural model, we mean topological information of the components rather than shape or form. Although such a task is in principle difficult because the search space of function is huge, the ontology plays a role to limit the search space. The ontology provides such primitives in the functional space that are the targets of the mapping, and screens out meaningless functional interpretations. The process of functional structure understanding consists of the following three steps: (1)behavior-function mapping, (2)identification of meta-functions among base-functions and (3)generation of functional hierarchies. See (Kitamura et al. 2000) for the detail of the identification of functional structures.

The first step is feasible because the search space spanned by FTs is four dimensional(Obj-Focus, O-Focus, P-Focus and Necessity) and each variable(an FT) takes only small number of discrete values, therefore, the system can exhaustively generate all the possible FT values, which we call interpretation, to map the behavioral model to a functional concept. When we assume the exhaustiveness of the functional terms(concepts) prepared, which is not unrealistic, we can eliminate such an interpretation that cannot correspond to any of the functional concepts. Of course, those that have a corresponding functional concept are not always meaningful in the context specified by the whole device structure, but they can be candidates for further investigation. The next step is identification of meta-functions among base functions extracted in the first step. In this step, those which has no meta-function associated with it are eliminated, since those have no contribution to any other base function. The last step is to consolidate each base function to form an *is-achieved-by* hierarchy of base functions. Meta-functions specify

cohesiveness between base functions by which we can combine two base functions into a macro-function. Because this way of combining two base functions are less dependent of structural information, functional structures different from structure-based functional hierarchies can be built. Without meta-functions, we need functional decomposition patterns appropriate for interpretation of the functional structure of the artifact under consideration in advance. The method using meta-functions, however, enables us to do the job without knowing such domain-dependent functional knowledge, which scales up the understanding system. The method has a set of general heuristics to build reasonable functional structures of variety. The understanding system has been implemented and evaluated using examples of a power plant and an oil-refinery plant.

## 7. Systematic Description of Functional Knowledge and Its Application

This section describes a more detailed example of the use of systematized knowledge.

### 7.1 Background

A designer often decomposes a required function into sub(micro)-functions, so-called functional decomposition (Pahl & Beitz 1988). As the result, a designer obtains a micro-macro hierarchy of functions, which represents that how the required (macro-)function is achieved by sub(micro)-functions, as a conceptual skeleton of the product realizing the requirement. Because there are many methods to achieve a specific function in general, designers should select an appropriate one from many alternatives. Such activity requires knowledge of how to achieve a function, which represents achievement relations among functions. We call such knowledge *functional knowledge*. Because many inventions have been realized by combination of techniques well-known in different domains (Sushkov et al. 1995), wideness of the coverage and variety of such functional knowledge in different domains is expected to facilitate innovative design. In order to provide designers with a wide range of functional knowledge, its systematic description in different domains is needed.

However, the design know-how including such functional knowledge used in the conceptual design phase is usually left implicit. Even if such knowledge is found in technical documents, it is often scattered around technical domains and improperly categorized. For example, a categorization of connection-methods found in a textbook published by an academic society is inconsistent and ill structured, since it is categorized according to different properties. Moreover, the textbook includes many categorizations based on the non-fundamental characteristics which can be derived from deeper principles.

We identify the following three reasons why we see such domain-dependency and such an inappropriate categorization of functional knowledge.

- 1) The first one is that different frameworks (viewpoints) for conceptualization are used when people try to describe knowledge in different domains.
- 2) The second one is several functional concepts are left undefined.
- 3) The last one is that structure (organization) of consistent functional knowledge is not fully investigated yet.

Aiming at systematization of functional knowledge for synthesis, we have discussed such ontologies that guide conceptualization of artifacts from the functional point of view thus far. We have proposed an extended device ontology and a functional concept ontology. The main role of such ontologies is to provide well-defined concepts for description of knowledge, and then to give a basis for systematization of knowledge.

This section discusses description of functional knowledge based on those functional ontologies. Firstly, we discuss *way of functional achievement* in detail. Next, we discuss development of a knowledge-based system to help human designers redesign an existing artifact. We focus on redesign not at the behavioral level but at the functional level in the conceptual design phase, in which a function is achieved using an alternative way satisfying given requirements. The supporting system includes three major sub-systems: (1) functional understanding system that identifies a functional structure of an artifact from given behavioral and structural models, (2) functional knowledge server that provides human designers with various ways of achievement, and (3) redesign solution proposing system that modifies the original functional structure identified by the functional understanding system in order to satisfy the given requirement. This section discusses the later two subsystems. The functional concept ontology and the systematized functional knowledge play crucial roles in these systems.

### 7.2 Framework of functional models of artifacts

In our functional modeling framework, the model of artifacts is summarized as shown in Figure 7. According to the extended device ontology, “structure” of the behavioral model describes the existence of components,



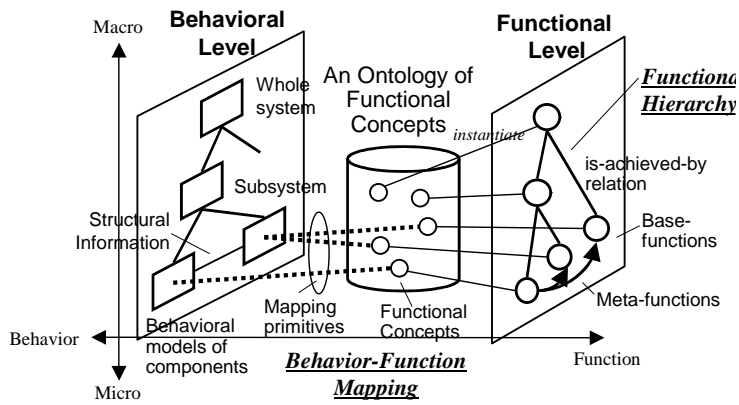


Figure 7: Framework of functional models of artifacts

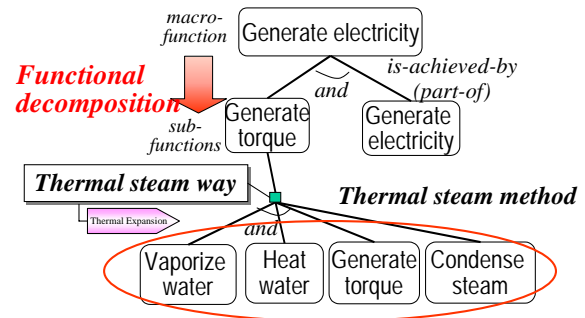


Figure 8: A functional hierarchy of a power plant (portion).

topological connections among them, and a micro-macro hierarchy among components and sub-systems. The behavioral model of components represents *B1 behavior* discussed in 4.4.3. The structural micro-macro relations represent hierarchical organization of behavior. On the other hand, at the functional level, a (base-) function of a component is defined as a result of teleological interpretation of a *B1 behavior* of the component under an intended goal. Each of the base-functions plays a role for another function (called meta-function) which represents interdependency among base-functions. The micro-macro relations among functions represent that the macro-functions are achieved by sequences of sub(micro)-functions (called functional achievement relations or functional decomposition discussed above. We also call such a hierarchy a functional hierarchy). The next section discusses this relation in detail.

### 7.3 The general knowledge of ways of functional achievement

As is discussed in 5.2, the key idea of our systematization of knowledge of functional decomposition is introduction of the idea of *way of functional achievement*. In Figure 8, for example, the basis of the second functional decomposition can be represented as “thermal steam way” whose intended phenomena is (adiabatic) heat expansion of gas. Such a way of achievement of a function can represent partial information of behavioral and/or structural one.

The key points of our systematization include: (1)the concept of *way*, (2)common functional concepts provided by the functional concept ontology, and (3)categorization of the functional knowledge. The first one enables us to explicate the background knowledge of functional achievement and thus gives us key concepts for organization of the knowledge, which realizes consistent categorization. The second one provides us common vocabulary for representation and then makes the functional knowledge reusable. The last one enables us to distinguish the inherent one from derivable ones, and thus makes the knowledge well-structured. The details are discussed in the following.

#### 7.3.1 Description of a way of achievement

The knowledge of achievement of a function is described as a set of ways of functional achievement. A description of a way of achievement of a function consists of:

- (1) the function as a (macro-)function
- (2) a set of sub(micro)-functions
- (3) temporal and causal constraints among sub-functions
- (4) principle of the achievement, conditions for use of the way, and
- (5) characteristics of products using the way.

The macro- and micro-functions are described in terms of general functional concepts which are provided by the functional concept ontology discussed in Section 5.4. Each of the provided functional concepts has general meaning of a base-function independent of target objects and components. Thus, the knowledge described in terms of them has a high possibility of application to other domains.

The principle of achievement represents physical principle or physical phenomena as justification to achievement of the macro-function. It governs the characteristics of the product using the way. For example, in the chemical way for connection, the chemical cohesiveness determines that the product cannot be disassembled without disrupt. The characteristics are described in a qualitative manner.



### 7.3.2 Categorization of knowledge of functional decomposition

We categorize typical representations of the knowledge into inherent knowledge based on their principles and the other representations derivable from the inherent one as follows. Figure 9 shows the categorization and their examples.

**The *is-a* hierarchy of ways:** The ways of achievement of a function are organized as an *is-a* hierarchy according to their principles. Because the principles are inherent properties of ways, we can recognize it as a right *is-a* hierarchy. For example, in Figure 9, ways of connection are categorized first into mechanical, chemical, and physical ways according to the principles for constraints of positions followed by further specialization. Other kinds of knowledge can be derived by reorganizing this *is-a* hierarchy. Note here that we have two types of *is-a* hierarchies in our functional ontologies: One is that of functional concepts which represents abstraction of functions themselves, that is, what goal is achieved and the other is that of way of functional achievement which represents abstraction hierarchy of the inherent properties such as principles of how to achieve a function. In our ontologies, we have only one hierarchy of the former for each domain of function(Energy, Object, Information and Force&motion as shown in Figure 5), while we have as many the latter hierarchies as functions, for example, each of *to connect*, *to exert force*, *to generate torque*, etc. can be the root of the hierarchy. Note also here that the latter *is-a* hierarchies should not be confused as a functional decomposition hierarchy which is described below.

**Attribute tree:** For selection of ways, ways are classified according to their attributes in a form of so-called a decision tree. Each leaf node represents a way. The structure of the tree depends on the viewpoints. In Figure 9, an attribute tree from the viewpoint of disassembly is shown. From the viewpoint, the important characteristics of a product using a specific way of connection include whether the product can be disassembled without disrupt or not and whether it can be done without deformation or not. Such characteristics appear as intermediate nodes in the attribute tree. Each of the ways of connection is classified by values of such characteristics and appears as a leaf node. This type of trees is sometimes confused with *is-a* hierarchy. They are the same in the sense that both are classification of ways, but are different from each other in that while the structure of the decision tree can be changed according to the purpose of the classification of interests and the intermediate nodes do not have to correspond to meaningful and reasonable categories of ways, the structure of *is-a* hierarchy is determined by its inherent property and hence it is unique. Confusion of this difference has been one of the causes of the inappropriate organization of way knowledge we have used to date.

**Functional decomposition tree (functional hierarchy):** A kind of a description (model) of a product at the functional level like one shown in Figure 7. It consists of sub-functions and description of the way as shown in Figure 8. This tree is constructed as a result of decision of the product designer.

**General functional decomposition tree:** It consists of all possible ways of achievement of a function in *OR* relationship. It is a kind of functional decomposition trees independent of a specific device. This type of tree is not stored in the system as it is. Like attribute trees, it is also generated by connecting each piece of functional concepts organized in an *is-a* tree upon request. All the *OR* branches in a path from the root to a leaf node in a tree of this type collectively define a *composite way* which is combination of some primitive ways defined in the *is-a* hierarchy as shown in Figure 9. In general, the ways described in textbooks are often composite one. For example,

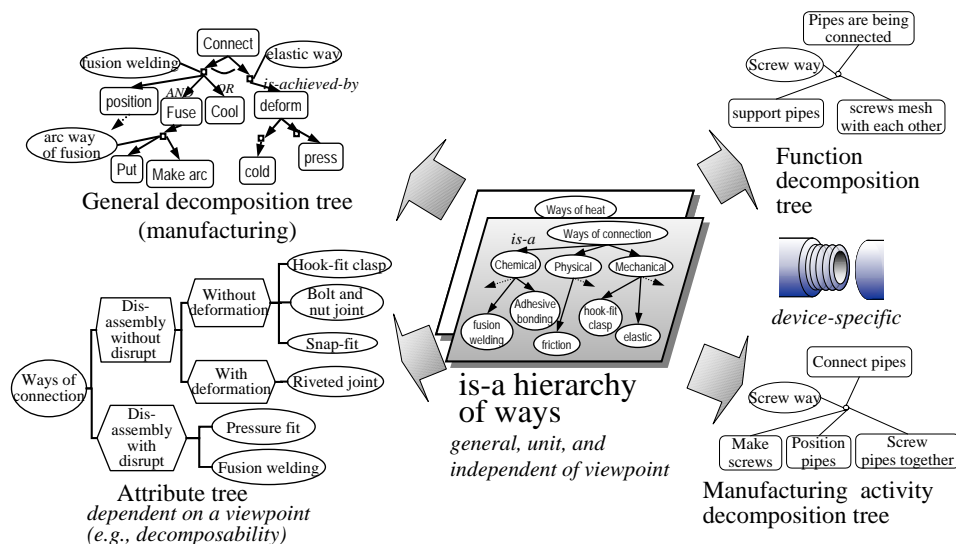


Figure 9: Categories of knowledge of ways and their examples

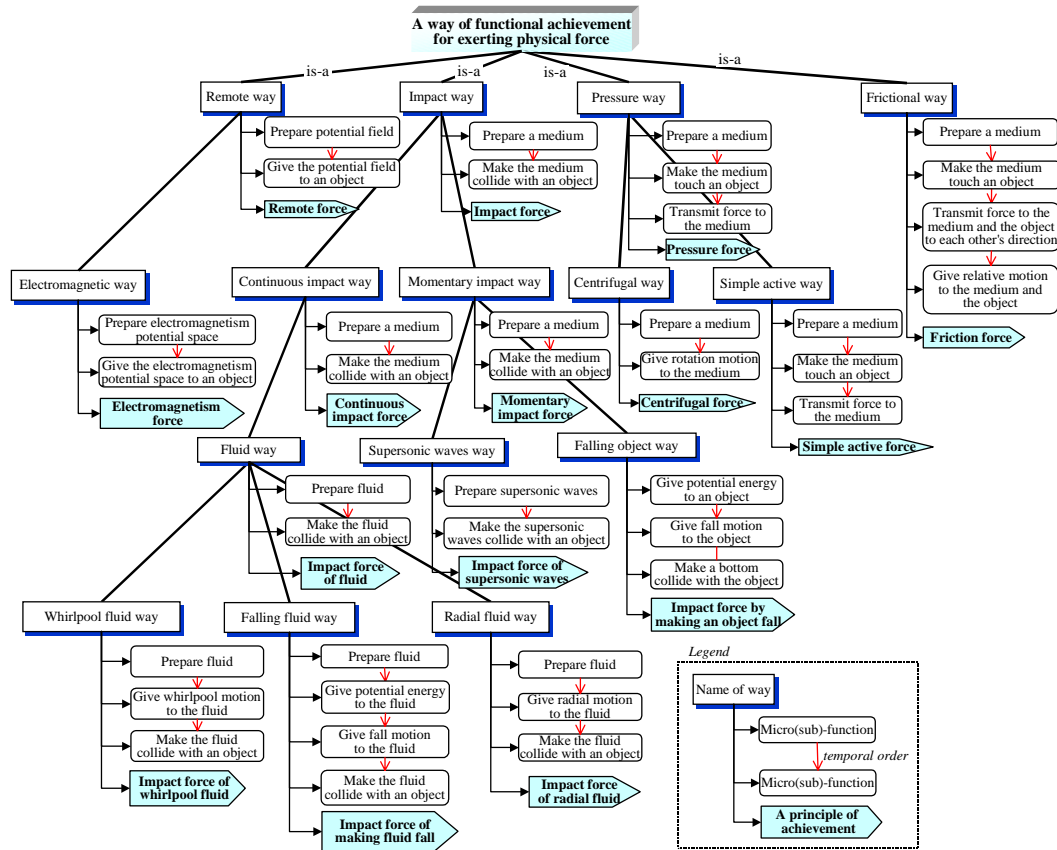


Figure 10: The knowledge base of ways of achievement for “to exert physical force” function (*is-a* hierarchy)

“the arc welding way” shown in a textbook is a composite of two primitive ways; “the fusion welding way of connection” and “the arc way of fusion” as shown in Figure 9. Note that the second way is not the way of connection but of fusion which is a micro-function of connection. These composite ways often cause an inappropriate structure of conventional organization of way knowledge. In our framework, they can be properly described as such composite ways in the general functional decomposition trees.

**Manufacturing activity decomposition tree:** It consists of manufacturing activities (process) of a product. This tree of a product and the functional decomposition tree of the product are usually different from each other. In the case of static functions such as connection, however, the principles of the ways in the both trees are often the same. Note that manufacturing activities of a product can be recognized as functions of a manufacturing device for the product. Thus, the common functional concepts can be used in the both trees.

**General manufacturing activity decomposition tree:** It consists of possible ways of manufacturing of a product in OR relation as the General functional decomposition tree.

### 7.3.3 Building knowledge base of ways of functional achievement

We have described 104 ways of achievement of 26 functions found in five devices in different domains: a washing machine, a printing device, slicing machines for ingot of semiconductors (using wire or rotating blade), and an etching device. Firstly, we described functional decomposition trees of them. Next, we generalized the ways that appear in the examples. Then, we tried to find their principles and then organized them into *is-a* hierarchies as discussed above. Lastly, we added other ways based on principles.

As an example, Figure 10 shows an *is-a* hierarchy of ways of achievement for “exerting physical force” and functional decomposition for each function. In the figure, a box, a round box, and a pentagon represent a concept of way, a sub(micro)-function of the macro-function in the way, and a principle of the way, respectively. The ways for exerting physical force are categorized into four types according to types of force; *remote force*, *impact force*, *pressure force*, and *friction force*. The *impact way* is further categorized into sub-types: the *continuous impact way* and the *momentary impact way* according to length of the time interval of application of force.

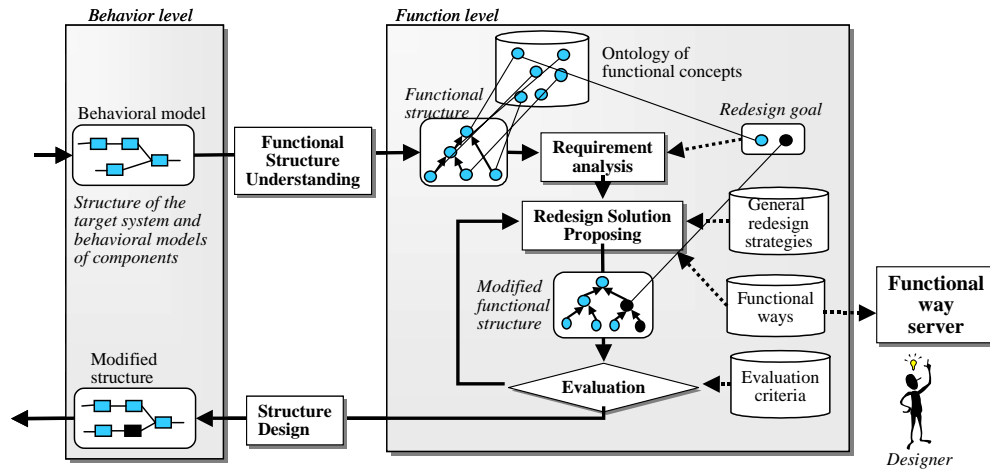


Figure 11: Overview of our intelligent redesign support system

All the sub-functions shown in Figure 10 can be defined in the same way as each function defined in Figure 10 to have its own sub-functions and to form other is-a tree of ways of achievement of other functions. The resulting functional decomposition patterns(relations) defined in Figure 10 can be used to form a functional decomposition tree of a specific object or a general functional decomposition tree as discussed in the previous section. Thus, *is-a* hierarchies of ways of achievement can generate “*is-achieved-by*” relations (a kind of *part-of* relations) among functions.

The knowledge of ways of achievement is general and hence applicable to systems in many different domains. In fact, the ways shown in Figure 10 are commonly used in some of the five examples. For example, many of washing machines use the *fluid way* for exerting force to cloth, while in the slicing machine the same way is also used in order to remove cutting dust.

#### 7.4 Application of the knowledge of achievement ways

In this section, we discuss development of a design supporting system based on the systematized functional knowledge discussed thus far to demonstrate how to use the systematized knowledge. The system is designed to help a human designer redesign an existing artifact at the functional level in the conceptual design phase. The supporting system as shown in Figure 11 includes three major (sub-)systems: a functional structure understanding system, a functional knowledge server, and a redesign solution proposing system. The functional concept ontology and the systematized functional knowledge play crucial roles in these systems as follows.

The first system, overviewed in 6.3, is designed to identify functional structures of artifacts from given behavioral models of components and their connection information (called functional structure understanding). This sub-system is needed as the first step of the redesign process in order to bridge the gap between the given behavioral level and the functional level for reasoning. The functional concept ontology enables the understanding system to make the search in the functional space tractable and to screen out meaningless interpretations. For more detail, see (Kitamura et al. 2000).

The identified functional structures contribute to innovative redesign. Given a functional structure of an existing artifact and a new requirement, to adopt another alternative way of functional achievement enables us to change the original functional structure into a different structure satisfying the requirement with possible drastic improvements.

Aiming at such innovative redesign, we developed the other two systems. The functional knowledge server is designed to provide human designers with various ways of achievement of the required function. On the other hand, the redesign solution proposing system tries to modify the original functional structure identified by the functional understanding system in order to satisfy the given requirement. We discuss these two systems in the following sections.

##### 7.4.1 Functional way server

When designers decompose the required function into sub-functions by selecting a “way” of achievement as mentioned above, the criteria for the selection depend on his/her own viewpoints and thus are different from each other. For example, a designer may adopt the “screw way” to realize the function “to connect two objects. The

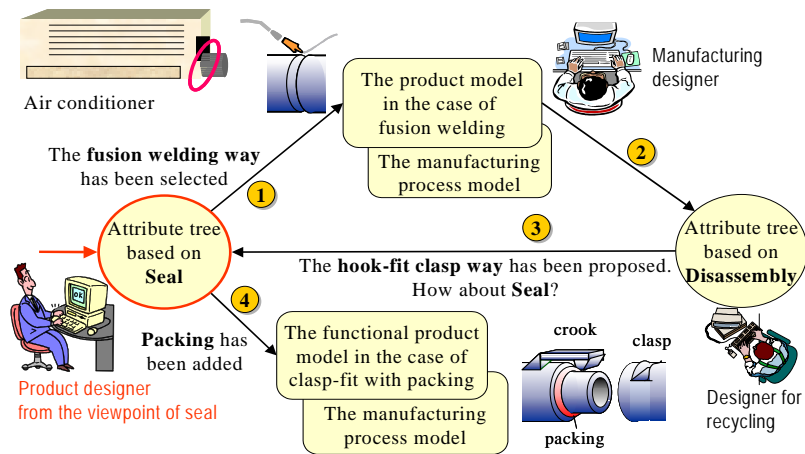


Figure 12. A scenario of use of the functional way server.

other designer, however, may prefer “hook-fit clasp way” in which the objects are connected by “to hook the crook” from the viewpoint of manufacturing (assembling) of the product, because manufacturing in the case of “hook-fit clasp” is easier than that of the “screw way”. In conventional knowledge description, such selection criterion and dependency on viewpoints are implicit. As discussed in Section 7.4.2, such knowledge dependent on viewpoints should be derived from the viewpoint-independent inherent knowledge.

The functional knowledge server supports conceptual design of engineering devices, providing suitable ways of achievement of the function that designers consider. The server contains the knowledge of ways of functional achievement of as *is-a* hierarchies. Given a required function and a viewpoint of the designer, the server reorganizes the knowledge structure suitable for the viewpoint and then shows ways in the form of an attribute(decision) tree. A viewpoint represents a context of designer’s thinking process. It consists of a phase in the product lifecycle such as manufacturing, using (product design) and recycling, focused attribute set, grain size, and a domain of interest such as mechanical and electrical.

Here we show a possible scenario of design using the functional knowledge server(See Figure 12). Let us suppose a concurrent design team is designing a connection between an air-conditioner and a pipe. Firstly, a product designer A looks for a way of connection from the viewpoint of seal in order to seal the coolant in the pipe. Given the viewpoint “seal”, the server looks up suitable attributes such as “joint gap” and “water resistance” associated with the viewpoint. According to the values of such attributes, the server classifies suitable ways, then shows the designer A an attribute-tree shown in Figure 13a. If the product designer selects the “fusion welding” way, a functional decomposition tree and a manufacturing activity decomposition tree are constructed (Figure 13b). A manufacturing designer can decompose the manufacturing-activity by selecting a way of fusion, say, the arc way.

Next, suppose that another designer B checks the connection from the viewpoint of recycling. The server derives the attribute tree shown in Figure 13c from the viewpoint of disassembly using the same knowledge of connection ways. If the designer B submits an alternative plan using “to hook-fit clasp”, the product designer A would recognize the characteristics of “hook-fit clasp” on seal in Figure 13a and thus add a sealant in order to seal the coolant.

The server provides a wide range of ways in different domains and then facilitates innovative design. Moreover, because the server can track effects of decisions made by both the product designer and the manufacturing designer, the server facilitates negotiation in a concurrent design team.

#### 7.4.2 Redesign solution proposing system

Redesign solution proposing system is designed to propose new functional structures of an existing artifact as a set of redesign solutions. The inputs for the system are the original functional structure identified by the functional understanding system and a new requirement given by users such as additional function or inconvenience to remove. As shown in Figure 12, the system uses the ways of functional achievement discussed thus far and redesign strategies. The former knowledge enables the system to modify functional structure drastically. The latter knowledge is concerned with the design process and provides sub-tasks in order to achieve redesign goals (Kitamura & Mizoguchi 1999b). The redesign strategies include the “alternative way” strategy in which ways of achievement in the original design are replaced by different ones. We focus this strategy here and discuss its sub-tasks in the following sub-sections.

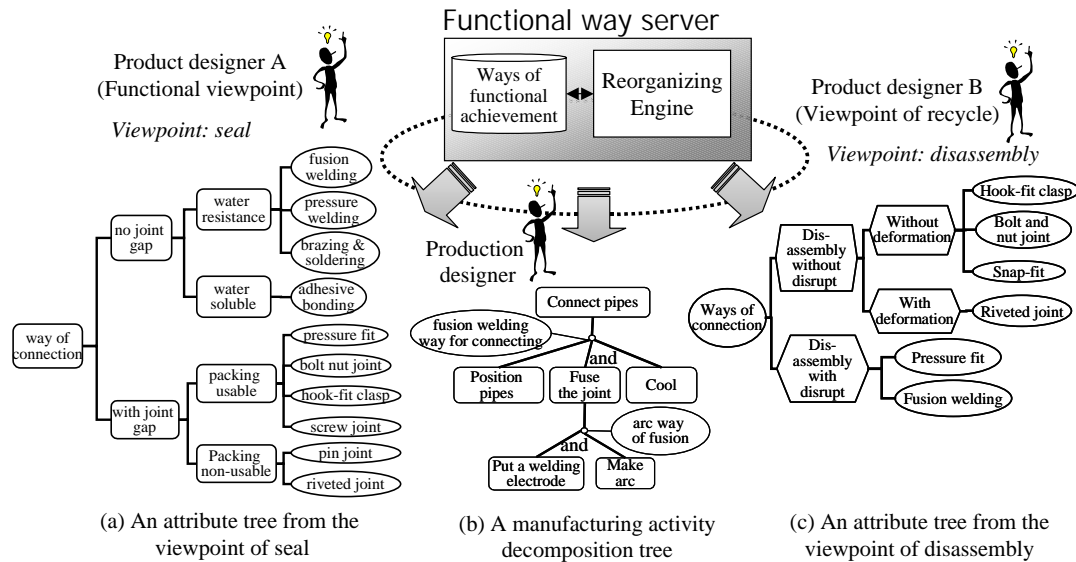


Figure 13: Framework of the functional way server and a scenario

### Steps of replacing ways of achievement

The steps of the redesign process based on the “alternative way” strategy is as follows:

Given an original functional structure and an inconvenience,

- Identify such ways that might cause the inconvenience (called the causative way)
- Enumerate possible alternative ways which can achieve the same macro-function or micro-functions of the causative way and have a possibility of removing the inconvenience.
- Evaluate the alternative ways from some viewpoints and show human designers the alternatives.
- Given the user’s choice, modify the original functional structure by replacing the causative way by the selected way and show it to the designer.
- Concerning other parts of the new functional structure, find possible violations of constraints of ways and/or possible improvement. If found, back to the first step.

The main idea of this strategy is to replace an original way of achievement having causative attribute (characteristic) of the inconvenience by new one having no such characteristic. The description of the way of achievement provides such characteristics in qualitative manner and constraints for use of the ways. The *is-a* hierarchy of the ways of achievement also enables the redesign system to propose new functional structures at some abstraction levels.

### Example of replacing ways of achievement

Figure 14 shows a given functional structure of a type of a washing machine. Here, the top-most function “to separate dirt from cloth” is achieved by “to lose combination force” and “to part dirt” (named “the way by weakening combination force”). The former micro-function is achieved by either “the way by formation change” or “the way by relative motion”. The latter micro-function is also achieved by “the way by relative motion”, whose both micro-functions are achieved by either “the whirlpool fluid way” or “the frictional way” in Figure 10. The friction force is caused also by whirlpool fluid made by a rotating screw.

Given the functional structure shown in Figure 14 and damage of cloth as an inconvenience, the system firstly identifies “the frictional way” as a causative way of the damage of cloth, since the description of the way includes damage of objects caused by friction. Next, the system enumerates such alternative ways for exerting force shown in Figure 10 that have principles without damage of objects. If a user selects “the centrifugal way” among the shown alternatives, the system constructs a new functional structure.



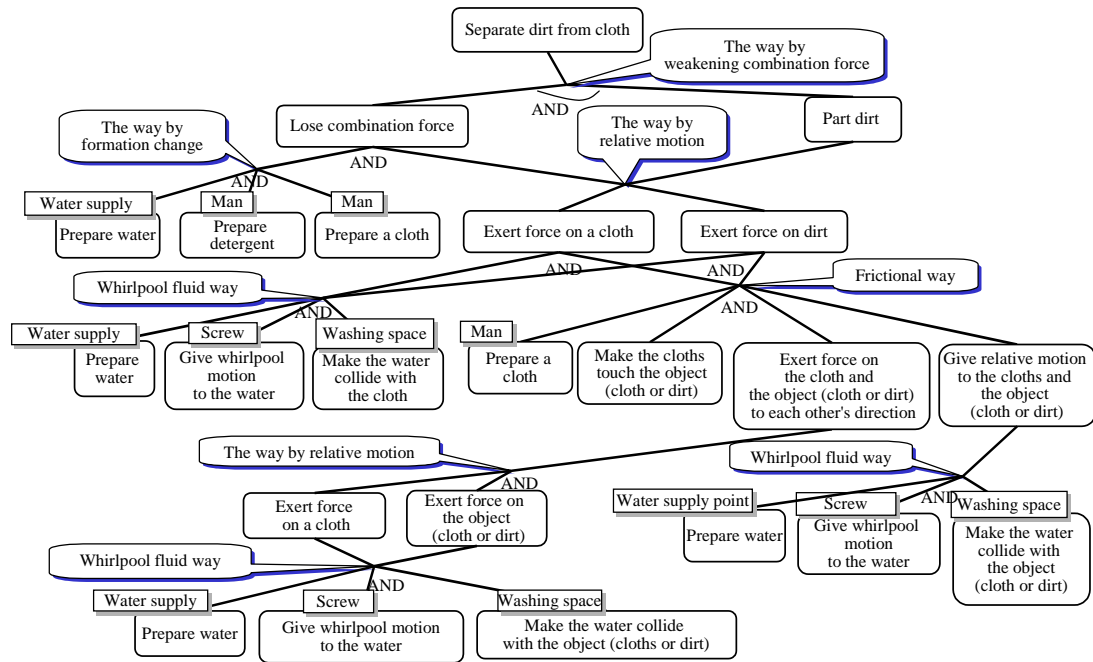


Figure 14: The original functional structure of the screw-type washing machine

Furthermore, the system tries to find rooms for further improvement. In this example, “the whirlpool fluid way” is found, because it uses the screw which became out of use in “the centrifugal way”. If a user decides to redesign this part, the system shows “the radial fluid way” as an alternative way, because this way has the same direction of force as “the centrifugal way”. When the system constructs a new functional structure again, it discovers that a constraint of “the way by relative motion”, which is the macro-function of “the radial fluid way”, is not satisfied. This constraint is that a force direction of the two sub-functions must be different. In order to get this constraint satisfied, the system suggests application of “the way by reactive force” for “exerting force on the cloth”. In this manner, the redesign by replacing ways of achievement is performed and then the new functional structure shown in Figure 15 is constructed.

This type of a washing machine has been on sale recently. Such modification enables us a drastic improvement, in this case, the reduction of damage of cloth caused by the friction. The system can propose other functional structures including one using “the falling object way” (so-called drum-type washing machine) and one using “the supersonic waves way” (this type also has been on sale recently).

The reasoning engine for redesign support has been implemented using Prolog (the prolog language running on Java platform called MINERVA). The inputs of the system are the original functional structure like Figure 14 and the ways of achievement like Figure 10. They are output of Hozo system (see Section 5.4) as an authoring environment for functional knowledge. One of the functional structures generated by the implemented system is depicted in Figure 15 (The current implementation does not support the function of diagram drawing).

## 8. Related work

### Definition of function:

Although there are quite amount of research on functional representation, definition of function is not established yet. In his early work on teleological analysis, de Kleer (1984) defines function as a causal pattern between variables. In many of other researches, function is defined as a kind of a desirable state or behavior intended (Chandrasekaran et al. 1993; Lind 1994; Umeda et al. 1996). There is no clear distinction between operational constituents of function and those of behavior. Especially, in FR (Sembugamoorthy & Chandrasekaran 1986), function is an intended desirable state of a device and behavior is defined as sequence of states and/or functions of one-level smaller grained components of the device. That is, the difference between function and behavior is relative depending on the grain size of description. Chandrasekaran and Josephson (1996) propose a representation of functions as effect to outside environment. Our definition of function is “teleological interpretation of (B1-)behavior” and is operationalized by introducing FT: functional topping. Unlike all the other definitions, our idea of function can explain its goal (context)-dependency and that multiple functions are derived from a behavior

according to the context.

### **Definition of functional concepts:**

The conventional functional knowledge uses “vocabulary” representing functionality (Sembugamoorthy & Chandrasekaran 1986; Umeda et al. 1996). Nevertheless, the meaning of terms in such vocabulary is implicit and hence, the contents of such knowledge depend on the modeler and thus tend to be ad hoc. Although some researches are done on computational definition of functions, the definitions are about an individual(instance) component and hence we find few generic discussion on functional concepts. In other words, the conventional functional concepts are rather ad-hoc and dependent on specific implementation of the component. This is why we often see inconsistent descriptions of functional concepts and cases where different labels and definitions are given to a single function.

On the other hand, value engineering (Tejima et al. 1981) has developed a standardized set of verbs representing function. Although it is well-organized, definitions of verbs are only for human consumption. Although a small set of functional concepts have been designed (Pahl & Beitz 1988; Chittaro et al. 1993; Lind 1994), they are at a very high level of abstraction and inflexible so that it is hard to incorporate designer’s intention into the functional model. Our definition of functional concepts has an explicit computational mechanism, FT: functional topping, for incorporating the way of interpretation of the behavior consistent with the intention of a designer.

The features of our definitions of functional concepts can be summarized as (1) use of mapping primitives with behavior and (2) independence of implementation. By the former, we mean that the functional concepts are defined by additional information to behavioral models using functional toppings (FTs) as primitives and then the information specifies the mapping from behavior to function. It enables us to ground the functional concepts on behavior and then to realize the behavior-function mapping as discussed in Section 6.3.

Concerning the latter point, the implementation of a function should be viewed as a composition of (a) behavior and structure dependent part and (b) behavior structure-independent decomposition of a function into sub-functions. To maximize the freedom of functional interpretation and then generality of functional concepts, the dependency should be minimized. Thus, we distinguish the behavior-function relation (a) from the “is-achieved-by” relation between functions (b). Although this distinction is shared with some researches (e.g., Lind 1994; Umeda et al. 1996), our definitions are independent from difference of realization. On the other hand, in FR (Sembugamoorthy & Chandrasekaran 1986), their functional models are described as sequences of partial states of the behavior, so that they depend on their implementations. Chandrasekaran and Josephson (1996) point out the importance of implementation-independent functional models and then propose representation of function as effect.

We also introduced a new type called meta-function. The CPD in CFRL (Vescovi, Iwasaki, Fikes & Chandrasekaran 1993) represents causal relations among functions. Lind (1994) categorizes such relations into Connection, Condition and Achieve. Rieger and Grinberg (1977) identify “enablement” as a type of the causal relation between states and action. The meta-functions are results of interpretation of such causal relations between functions under the role of the agent function for the target functions without mention of the objects associated with components.

### **Definition of and reference to function:**

Confusion between definition of function and reference to it is often seen in functional descriptions. In other words, what a functional concept means at all and how it is achieved are different and should not be confused. This is necessary for realizing implementation-independent definition of a functional concept, which is realized in our work.

### **Functional structure:**

An artifact can be interpreted to have multiple functional structures according to the viewpoints. The conventional research on it, however, can only produce, for each artifact, a single functional structure analog to its topological structure. Our method of functional structure derivation is based on meta-functions that introduce cohesiveness between functions independently of the device structure and heuristics that control the consolidation process to generate various functional structures including one analog to the device structure.

### **Ontology of artifacts:**

The importance of explicit specification of conceptualization (i.e., ontology) in artifact modeling is widely recognized in literature such as (Abu-Hannna & Jansweijer 1994; Borst, Akkermans & Top 1997; Horvath, Kucgozi & Vergeest 1998; Salustri 1998; Chandrasekaran & Josephson 2000). As pointed out these papers,

ontological statements (commitment) can be used as simplifying assumptions to improve the robustness, complexity, and computability of knowledge representation of artifact models, and to avoid misinterpretation of them.

Borst et al. (1997) propose an hierarchy of ontology for designing an artifact which shares a lot of with our work. The major differences include that Borst's ontologies do not include one for function which is our main issue and that his device ontology is weak in that it does not have enough concepts for understanding the roles of all the participants play comparing to our extended device ontology.

Bond graph is a theory for describing a system domain-independently in the field of system dynamics (Rosenberg & Karnopp 1983). It introduces *flow* that represents amount of something which flows in the device and *effort* which has capability to cause the *flow*. Bond graph represents the behavior of the device by combining each device model built in terms of *effort* and *flow*. Although it is elaborated well, it still lacks the expressive vocabulary and device ontology to specify viewpoint in modelling an artifact. In the flow-based functional modeling approaches (Chittaro et al. 1993; Lind 1994), primitive functions are proposed based on such generalized *flow* concept. They are included in our functional concept ontology based on the extended device ontology.

Horvath et al. (1998) discuss design concept ontologies for comprehensive methodology for handling design concepts in conceptual design, which include structure and shape as well as functionality. For example, in the structural view, the connected entities are specified by positional, morphological, kinematical, and functional descriptors. We concentrate on functionality and then categorize connections among devices according to their functions, that is, kinds of transmitting force or motion in the mechanical domain.

Chandrasekaran and Josephson (2000) try to clarify many meanings of the concept "function" and discuss relationship between environment-centric functions and device-centric functions based on ontological consideration. Although we share the distinction between the two major views of functions and the attitude towards the ontological analysis with them, we concentrate only on the device-centric viewpoint in this paper. We have explicated ontological meaning of concepts behind the viewpoint including roles played by entities. As a result, we have explained the ontological difference between a behavior and a function, while they consider a function is one of the intended behaviors of a device.

### **Ways of functional achievement:**

General knowledge about functional decomposition is extensively discussed in design community (Bradshaw & Young 1991; Bhatta & Goel 1997; Umeda et al. 1997). It superficially looks similar to our idea of way of functional achievement. Nevertheless, these two are largely different from each other. The major differences between the two include explicit description of "way", organization in *is-a* hierarchies based on principles of ways and a functional concept ontology as follows.

Firstly, our general ways of functional achievement explicitly represent the feature of achievement such as theory and phenomena. They enable the system to facilitate the smooth interaction between models at the structural and functional levels. The designer can check the feasibility of functional decomposition using the features represented as the ways as behavioral constraint.

Secondly, we organized such general knowledge as an *is-a* hierarchy. Although the feature knowledge is also captured in (Malmqvist 97), it corresponds to the functional decomposition tree of a specific product in our categorization shown in Section 7.4.2 and there is no organization of general knowledge. The crucial issue of organization as an *is-a* hierarchy is to capture inherent properties in order to avoid improper categorization which we sometimes see in the conventional knowledge organization. The concept of way enables us to capture the principles of achievement of a function (i.e., why the function can be achieved) as the inherent properties and thus to realize consistent categorization.

Such systematization of functional knowledge based on the concept of way and the functional concepts enables the redesign support systems using the functional knowledge to apply a wide range of alternative ways in different domains. Such knowledge in a different domain can facilitate innovative design many of which are based on techniques well-known in different domains (Sushkov et al. 1995). TechOptimizer (Invention Machine Corp. 1999) is a software product based on a theory for innovative design (Sushkov et al. 1995), which contains generic principles of invention. It, however, just searches highly abstract principles for given criteria and proposes no redesign solution of the target system, while our redesign proposing system modifies the original functional structure using the *is-a* (abstraction) hierarchy of the ways of achievement. Moreover, it is not adaptive for the



designers' viewpoints, while our functional way server reorganizes the general knowledge according to the designer's viewpoint.

## 8. Concluding remarks

We have discussed and designed layers of ontologies for functional modelling of an artifact aiming at supporting design knowledge systematization. We have shown that the extended device ontology contributes to consistent model building of artifacts and knowledge base building sharable across domains and to explication of the differences of functional knowledge in different domains which are seemingly incompatible with each other. The layers of ontologies thus help us systematize functional knowledge. Another contribution of this research can be summarized as a framework of systematization of design knowledge about functional decomposition. We proposed "way" of functional achievement as a key concept for systematization. The benefits of such functional knowledge in a concurrent design team were shown using a scenario. All the systems described in this paper have been implemented.

The authors have advocated the importance of "Content-oriented" research rather than form-oriented research that have dominated AI research to date. The research described in this paper is a result in this direction. Knowledge processing never loses its importance and requires more sophisticated treatment of knowledge rather than inference mechanisms. To cope with the high demand on advanced knowledge processing, we need in-depth understanding about the nature of knowledge and viewpoints to model the target world, framework of knowledge description supported by solid foundation of conceptualization, and so on. Ontology engineering for systematizing knowledge will become more important in the coming years.

## Acknowledgements

The authors would like to thank Toshinobu Kasai, Kouki Higashide, Toshio Ueda, Toshinobu Sano, Masaru Takahashi, Mariko Yoshikawa and Tomonobu Takahashi for their contributions to this work. The authors are grateful to Mitsuru Ikeda for his valuable comments. This research is supported in part by the Japan Society for the Promotion of Science (JSPS-RFTF97P00701). Special thanks go to the members, Professors. Eiji Arai, Masahiko Onosato and Hiroshi Kawakami. The functional way server mentioned in 7.4.1 is result of an international collaborative research project, GNOSIS within the Intelligent Manufacturing Systems (IMS) research program.

## Reference

- ABU-HANNA, A. & JANSWIJER, W. (1994), Modeling Domain Knowledge Using Explicit Conceptualization, *IEEE Expert*, 9(5), 53-63.
- BHATTA, S. R. & GOEL, A. K. (1997). A Functional Theory of Design Patterns, In *Proc. of IJCAI-97*, 294-300.
- BORST, P., AKKERMANS, H., & TOP, J. (1997). Engineering Ontologies, *Int'l Journal of Human-Computer Studies*, 46(2/3), 365-406.
- BRADSHAW, J. A & YOUNG, R. M. (1991). Evaluating Design using Knowledge of Purpose and Knowledge of Structure. *IEEE Expert*, 6(2), 33-40.
- CHANDRASEKARAN, B., GOEL, A. K., & IWASAKI. (1993). Functional Representation as Design Rationale, *COMPUTER*, 48-56.
- CHANDRASEKARAN, B. & JOSEPHSON J. R. (1996). Representing Function as Effect: Assigning Functions to Objects in Context and Out. In *Proc. of AAAI-96 Workshop on Modeling and Reasoning with Function*.
- CHANDRASEKARAN, B. & JOSEPHSON J. R. (2000). Function in Device Representation, *Engineering with Computers*, 16(3/4), 162-177.
- CHITTARO, L., GUIDA, G., TASSO, C. & TOPPANO, E. (1993). Functional and Teleological Knowledge in the Multimodeling Approach for Reasoning about Physical Systems: A case Study in Diagnosis, *IEEE Transactions on Systems, Man, and Cybernetics*, 23(6), 1718-1751.
- DE KLEER, J. & BROWN, J. S. (1984). A Qualitative Physics Based on Confluences, *Artificial Intelligence*, 24, 7-83.
- DE KLEER, J. (1984) How Circuits Work., *Artificial Intelligence*, 24, 205-280.
- FORBUS, K. D. (1984). Qualitative Process Theory, *Artificial Intelligence*, 24, 85-168, 1984.
- HODGES, J. (1992). Naive mechanics - a computational model of device use and function in design improvisation, *IEEE Expert*, 7(1):14-27.
- HORVAH, I., KUCZOGLI, GY. & VERGEEST, J. S. M. (1998). 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.
- INVENTION MACHINE CORP. (1999). TechOptimizer, <http://www.invention-machine.com/products/techoptimizer.cfm>.

- KEUNEKE, A. M. (1991). A. Device Representation: the Significance of Functional Knowledge, *IEEE Expert*, 24, 22-25.
- KITAMURA, Y., & MIZOGUCHI, R. (1999a). Meta-Functions of Artifacts, *Proc. of The Thirteenth International Workshop on Qualitative Reasoning (QR-99)*, 136-145.
- KITAMURA, Y., & MIZOGUCHI, R. (1999b). Towards Redesign based on Ontologies of Functional Concepts and Redesign Strategies, *Proc. of the 2nd International Workshop on Strategic Knowledge and Concept Formation*, pp.181-192.
- KITAMURA, Y., SANO, T. & MIZOGUCHI, R. (2000). Functional Understanding based on an Ontology of Functional Concepts, *Proc. of The Sixth Pacific Rim International Conference on Artificial Intelligence (PRICAI 2000)*, pp.723-733, Springer-Verlag.
- KOZAKI, K., KITAMURA, Y., IKEDA, M., & MIZOGUCHI R. (2000). Development of an Environment for Building Ontologies which is based on a Fundamental Consideration of "Relationship" and "Role", *Proc. of The Sixth Pacific Knowledge Acquisition Workshop (PKAW2000)*, 205-221.
- LIND, M. (1994). Modeling Goals and Functions of Complex Industrial Plants. *Applied artificial intelligence*, 8, 259-283.
- MALMQVIST, J. (1997). Improved function-means trees by inclusion of design history information, *Journal of Engineering Design*, Vol.8, No.2, pp.107-117.
- MILES, L. D. (1961). *Techniques of value analysis and engineering*. McGraw-hill.
- MIZOGUCHI, R., & IKEDA, M., Towards Ontology Engineering, *Proc. of The Joint 1997 Pacific Asian Conference on Expert systems / Singapore International Conference on Intelligent Systems*, pp. 259-266, 1997., also Technical Report AI-TR-96-1, I.S.I.R., Osaka University, <http://www.ei.sanken.osaka-u.ac.jp/pub/miz/miz-onteng.pdf>
- MIZOGUCHI, R. & KITAMURA, Y. (2000). Foundation of Knowledge Systematization: Role of Ontological Engineering, *Industrial Knowledge Management - A Micro Level Approach*, Rajkumar Roy Ed., Chapter 1, 17-36, Springer-Verlag, London.
- MORTENSEN, N. H. (1999). Function Concepts for Machine Parts - Contribution to a Part Design Theory, *Proc. of ICED 99*, 2, 841-846.
- PAHL, G., & BEITZ, W. (1988). *Engineering design - a systematic approach*, The Design Council.
- RIEGER, C., & GRINBERG, M. (1977). Declarative representation and procedural simulation of causality in physical mechanisms. In *Proc. of IJCAI-77*, 250-256.
- ROSENBERG R. C. & KARNOPP, D. C. (1983) Introduction to Physical System Dynamics. McGraw-Hill, 1983.
- SALUSTRI, F. A. (1998). 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*, 31-51.
- SASAJIMA, M., KITAMURA, Y., IKEDA, M., & MIZOGUCHI, R. (1995). FBRL: A Function and Behavior Representation Language, *Proc. of IJCAI-95*, pp.1830 – 1836.
- SEMBUGAMOORTHY, V. & CHANDRASEKARAN, B. (1986). Functional representation of devices and compilation of diagnostic problem-solving systems, In *Experience, memory and Reasoning*, 47-73.
- SUSHKOV, V.V. MARS, N.J.I., & WOGNUM, P.M.(1995). Introduction to TIPS: a theory for creative design, *Artificial Intelligence in Engineering*, 9.
- TAKEDA, H., VEERKAMP, P., TOMIYAMA, T. & YOSHIKAWA, H. (1990). Modeling design processes, *AI Magazine*, 11(4), 37-48.
- TEJIMA, N. ET AL. (eds). Selection of functional terms and categorization, Report 49, Soc. of Japanese Value Engineering (In Japanese), 1981
- TOMIYAMA, T. (2000). A theoretical approach to synthesis, *Proc. of 2000 International symposium on modeling of synthesis*, pp.25-64.
- UMEDA, Y., ISHII, M., YOSHIOKA, M., SHIMOMURA, Y., & TOMIYAMA, T. (1996). Supporting conceptual design based on the function-behavior-state modeler. *Artificial Intelligence for Engineering Design, Analysis and Manufacturing*, 10, 275-288.
- VESCOVI, M.; IWASAKI, Y.; FIKES, R.; & CHANDRASEKARAN, B. (1993). CFRL: A language for specifying the causal functionality of engineered devices. In *Proc. of AAAI-93*, 626-633.
- YOSHIKAWA, H. (1981). General design theory and a CAD system, *Man-machine Communication in CAD/CAM*, ed. By Sata, T. and Warman, E.A., North-Holland, Amsterdam, pp.35-58.