Foundation of Knowledge Systematization: Role of Ontological Engineering

Riichiro Mizoguchi and Yoshinobu Kitamura

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

Abstract: The objective of the research described in this article is to give a firm foundation for knowledge systematization from AI point of view. Our knowledge systematization is based on ontological engineering to enable people to build knowledge bases as a result of knowledge systematization. The basic philosophy behind our enterprise is that ontological engineering provides us with the basis on which we can build knowledge and with computer-interpretable vocabulary in terms of which we can describe knowledge systematically. The ontology we developed includes device ontology and functional ontology both of which play fundamental roles in understanding artifacts by a computer. Redesign of a manufacturing process through its functional understanding in industrial engineering is taken as an example task to show how to use the ontologies as well as the way of knowledge systematization of manufacturing processes. A sample ontology of a manufacturing process is also built in terms of functional ontology from device ontology point of view.

1. Introduction

In AI research history, we can identify two types of research. One is "Form-oriented research" and the other is "Content-oriented research". The former investigates formal topics like logic, knowledge representation, search, etc. and the latter content of knowledge. Apparently, the former has dominated AI research to date. Recently, however, "Content-oriented research" has attracted considerable attention because a lot of real-world problems to solve such as knowledge reuse, facilitation of agent communication, media integration through understanding, large-scale knowledge bases, etc. require not only advanced formalisms but also sophisticated treatment of the content of knowledge before it is put into the formalisms.

Formal theories such as predicate logic provide us with a powerful tool to guarantee sound reasoning and thinking. It even enables us to discuss the limit of our reasoning in a principled way. However, it cannot answer any of the questions such as what knowledge we should prepare for solving the problems given, how to scale up the knowledge bases, how to reuse and share the knowledge, how to manage knowledge and so on. In other words, we cannot say it has provided us with something valuable to solve real-world problems.

In expert system community, the knowledge principle proposed by Feigenbaum has been accepted and a lot of development has been carried out with a deep appreciation of the principle, since it is to the point in the sense that he stressed the importance of accumulation of knowledge rather than formal reasoning or logic. This has been proved by the success of the expert system development and a lot of research activities has been done under the flag of “knowledge engineering”. The authors are not claiming the so-called rule-base technology is what we need for future knowledge processing. Rather, treatment of knowledge should be in-depth analyzed further to make it sharable and reusable among computers and human agents. Advanced knowledge processing technology should cope with various knowledge sources and elicit, transform, organize, and translate knowledge to enable the agents to utilize it.

Although importance of such "Content-oriented research" has been gradually recognized these days, we do not have sophisticated methodologies for content-oriented research yet. In spite of much effort devoted to such research, major results were only development of KBs. We could identify the reasons for this as follows:

1) It tends to be ad-hoc, and

2) It does not have a methodology which enables knowledge to accumulate.

It is necessary to overcome these difficulties in order to establish the content-oriented research.

Ontological Engineering has been proposed for that purpose [1]. It is a research methodology which gives us design rationale of a knowledge base, kernel conceptualization of the world of interest, semantic constraints of concepts together with sophisticated theories and technologies enabling accumulation of knowledge which is dispensable for knowledge processing in the real world. The authors believe knowledge management essentially needs content-oriented research. It should be more than information retrieval with powerful retrieval functions. We should go deeper to obtain the true knowledge management.

The major topic of this paper is the knowledge systematization necessary for an intelligent design support system. This is indeed a topic of content-oriented research and is not that of a knowledge representation such as production rule, frame or semantic network. Although knowledge representation tells us how to represent knowledge, it is not enough for our purpose, since what is necessary is something we need before the stage of knowledge representation, that is, knowledge organized in an appropriate structure with appropriate vocabulary. This is what the next generation knowledge base building needs, since it should be principled in the sense that it is based on well-structured vocabulary with an explicit conceptualization of the assumptions. This nicely suggests ontological engineering is promising for the purpose of our enterprise.

While every scientific activity which has been done to date is, of course, a kind of knowledge systematization, it has been mainly done in terms of analytical formulae with analytical/quantitative treatment. As a default, the systematization is intended for human interpretation. Our knowledge systematization adopts another way, that is, ontological engineering to enable people to build knowledge bases on the computer as a result of knowledge systematization. The philosophy behind our enterprise is that ontological engineering provides us with the basis on which we can build knowledge and with computer-interpretable vocabulary in terms of which we can describe knowledge systematically in a computer-understandable manner.

This paper summarizes the results of the research the authors have conducted to date on this topic and is organized as follows: The next section briefly describes ontological engineering. Section 3 presents the framework of knowledge systematization and function in design. Basic ontologies used as the foundation of knowledge systematization are discussed in Section 4. Section 5 discusses the main topic, functional ontology, followed by its applications to knowledge of manufacturing process in industrial engineering.

2. Ontological Engineering

An ontology is

an explicit specification of objects and relations in the target world intended to share in a community and to use for building a model of the target world.

Amongst its many utilities, it works as:

- **A common vocabulary.** The description of the target world needs a vocabulary agreed by people involved. Terms contained in an ontology contribute to it.
- **Explication of what has been left implicit.** In all of the human activities, we find presuppositions/assumptions that are left implicit. Typical examples include definitions of common and basic terms, relations and constraints among them, and viewpoints for interpreting the phenomena and target structure common to the tasks they are usually engaged in. Any knowledge base built is based on a conceptualization possessed by the builder and is usually implicit. An ontology is an explication of the very implicit knowledge. Such an explicit representation of assumptions and conceptualization is more than a simple explication. Although it might be hard to be properly appreciated by people who have no experience in such representation, its contribution to knowledge reuse and sharing is more than expectation considering that the implicitness has been one of the crucial causes of preventing knowledge sharing and reuse.
- **Systematization of knowledge.** Knowledge systematization requires well-established vocabulary/concepts in terms of which people describe phenomena, theories and target things under consideration. An ontology thus contributes to providing backbone of systematization of knowledge. Actually, the authors have been involved in a project on Knowledge systematization of production knowledge under the umbrella of IMS: Intelligent Manufacturing Systems project [2].
- **Design rationale.** Typical examples to be explicated include intention of the designers of artifacts, that is, part of design rationale. An ontology contributes to explication of assumptions, implicit preconditions required by the problems to solve as well as the conceptualization of the target object that reflects those assumptions. In the case of diagnostic systems, fault classes diagnosed and range of the diagnostic inference, in the case of qualitative reasoning systems, classes of causal relations derived, and so on.
- **Meta-model function.** A model is usually built in the computer as an abstraction of the real target. And, an ontology provides us with concepts and relations among them that are used as building blocks of the model. Thus, an ontology specifies the models to build by giving guidelines and constraints that should be satisfied in such a model. This function is viewed as that at the meta-level.

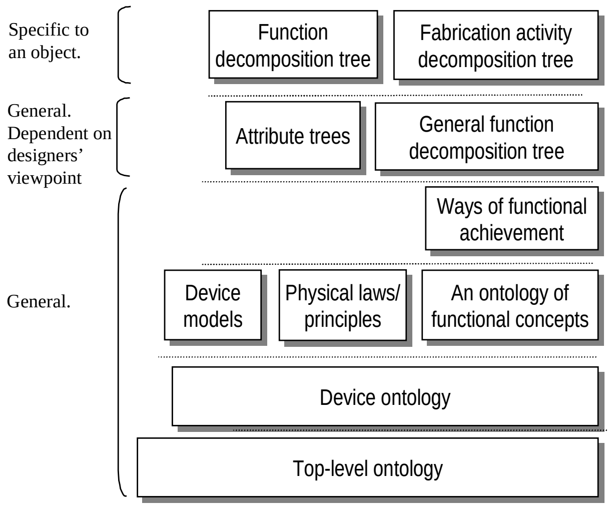
Specific to an object.
General. Dependent on designers' viewpoint
General.
Function decomposition tree
Fabrication activity decomposition tree
Attribute trees
General function decomposition tree
Ways of functional achievement
Device models
Physical laws/ principles
An ontology of functional concepts
Device ontology
Top-level ontology

Figure 1: Hierarchical organization of ontologies.

In summary, an ontology provides us with "a theory of content" to enable research results to accumulate like form-oriented research avoiding ad-hoc methodologies which the conventional content-oriented activities have been suffering from.

3. The Framework of Knowledge Systematization

3.1 Framework

Figure 1 shows the sketch of multi-layered ontologies for ontological knowledge systematization together with some kinds of functional knowledge. We utilize the two major functionalities of an ontology, that is, common vocabulary function and meta-model function as well as knowledge systematization functionality. The former is essential for describing knowledge. The latter is for formulating ontologies in a multi-layered structure to keep it in a moduler structure. Ontologies are arranged in up-side-down to stress the analogy of physical construction. An ontology in a layer plays the role of a meta-ontology of those in an upper layer. Top-level ontology stays at the bottom layer to govern all the ontologies in the upper layers. Next is the device ontology which is effectively the basis of our knowledge systematization. On top of device ontology, functional ontology is built together with ontology of ways of function achievement discussed in section 5.2 as above knowledge are organised. Further, several kinds of knowledge is organized in terms of the concepts in the ontologies. It is important to note here that the first four layers from the bottom contain **definitions** of concepts and other higher layers are for **description** of various kinds of knowledge in terms of concepts defined in the lower layers.

3.2 Function in design

No one disagrees 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. A top-level ontology is like one discussed in philosophy and usually consists of a few categories such as entity, relation, role, time, space, etc.

One of the key claims of our knowledge systematization is that the concept of function should be defined independently of an object which can possess it and of its realization method. Contrary to the possible strong disagreement from the people working in the engineering design, it 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. Another justification is reusability of functional knowledge. If functions are defined depending on objects and their realization, few function 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.

In the abstraction hierarchy of function, there are various functional concepts. At the higher levels, we find very general ones such as to transmit, to achieve, to drive, et al., at the lower levels, many domain-specific functions appear. Our interest includes to identify how many categories of function there are, in what relations they are one another, how those categories facilitate use of functional knowledge in design process, etc. Furthermore, we investigate the relationship between function and behavior. That is, a functional ontology

Function has been a major research topic in intelligent design community[3,4,5,6]. In many of the workshops held in IJCAI, AAAI and ECAI, workshops on functional modelling have been held almost every time. The following is a list of reasons why the concept of function is critical in design:

1. Most of the requirements are given in terms of functional concepts as is seen in the statement "Design is a mapping from functional description to structural one".
2. Understanding of functional terms is necessary for computers to interpret requirements specification.
3. In order to realize functions, computers have to know the relations among functions and behaviors.
4. A rich set of functional concepts enable a computer to make inference on functional space to find a solution.
5. Major portion of design rationale of design is left as functional description.
6. Philosophically, the concept of the "whole" of any artifact largely depends on its functional unity. Without functional concepts, we cannot explain why an artifact or a module is meaningful and complete as it is.

4. Basic Ontologies

Although our major objective in the current research is to establish a functional ontology for design knowledge systematization, we cannot avoid to mention other ontologies which give foundations of it. It is device ontology and top-level ontology discussed below.

4.1 Top-level ontology

This is the most fundamental ontology and provides necessary set of categories to talk about every thing in the world. It may include the following concepts:

```
-Substrate                         -Relation
  -Space                           -Role
  -Time                            -Attribute
  -Substance                       -Matter
-Entity                              -Event
  -Concrete things                   -Fact
    -Objects                         -State
    -Process                     -Representation
      -Action                      -p/o form: Representation form
      -Phenomena                   -p/o content: Matter
  -Abstract things             -Representation form
```

In spite of its importance, we do not go into details of the top level ontology here.

4.2 Device ontology

Concerning modelling of artifacts, there exist two major viewpoints: Device-centered and Process-centered views. The device ontology specifies the former and the process ontology the latter. 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 concentrates on phenomena occurring in each place(device) to obtain the output result with paying little attention to the devices existing there.

The major difference between the two is that while device ontology has an ***Agent*** which is considered as something which 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 the device in device ontology. It is natural to consider process ontology is more basic than device ontology. Before humans had started to make an artifact, only natural phenomena had been out there in the world. Any device and/or artifact is something built by limiting the environment where phenomena of interest occur to what the device provides and by connecting them to achieve the goal set in advance. Every device utilizes several natural phenomena to realize its functions.

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

1. Device ontology is straightforward because every artifact is a device.
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. Theoretically, process ontology cannot have function associated with it because it has no ***Agent***.
5. Device-centred view saves a load of reasoning about design, it hides internal details of devices.
6. 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.

It is important to note, however, device ontology is not almighty. Essentially, device ontology views any device as a black box. Of course, in the device ontology world, one can go into smaller grain world in which the parent device is seen as configuration of several sub-devices of smaller grain size. But, 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.

4.3 Constituents of device ontology

In this article, we basis all of our discussion on device ontology. Top-level categories of device ontology include ***Entity, Role, Structure, Behavior*** and ***Function***. ***Entity*** represents the real and basic things existing in the world such as human, device, fluid, solid, etc. ***Role*** represents roles the entity plays during the process where the device works. It has functional role and structural role. The former includes ***Agent*** which is an actor who does the action to cause the effect required, ***Object*** which is processed by the agent, ***Medium*** which carries something used by the device, ***material*** which is the main source of the results output from the device and ***product*** which is the regular output of the device and is made of/from the material. The structural role has sub-roles such as ***input, output*** and ***component*** which is the role played by any device to form the structure. Not all the ***input*** has material role, though all ***material*** have input role. Similarly, not all the ***output*** have product role, though all ***product*** have output role. ***Structure*** has sub-categories such as ***inlet***, ***outlet*** and ***connection*** which is a relational concept connecting devices. A component has inlets and outlets whose role is to connect it with others. Input (output) is anything coming into (going out of) a device through an inlet (outlet).

Behavior of a device (***DO-Behavior*** in short) is defined as "situation-independent conceptualization of the change between input and output of the device" and it has two subcategories such as ***transitive behavior*** and ***intransitive behavior*** which roughly correspond to transitive and intransitive verbs, respectively. Roughly speaking, the former is concerned with the change of the ***object*** and the latter with that of the ***agent***. Note here that we do not define behavior in general, that is, we only discuss behavior of a device. ***Function****(of a device)* is defined as "teleological interpretation of a ***DO-behavior***" [5].

We now would like to defend our definition of ***DO-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 numerical simulation, which is the typical way of simulation, by simulation of the behavior of a system(device), 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 of how a thing behaves which is not explained by DO-Behavior.

This view is parallel to the naïve view of behavior like the idea 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. This is what AI research needs about behavior of a system from device-centered point of view.

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. 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, DO-behavior which 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 “***BO-behavior***” for simplicity.

5. An Ontology of Functional Concepts

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 source of confusions in understanding what a function is. This view tells us every action itself can be a function. In design research, however, what we want is not such a view of 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. In the design context, speaking is a behavior and its function would be transmitting information in such a manner that a human can hear; walking is a behavior and its function would be carrying itself from one place to another or consuming energy.

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 already defined ***Function*** of a device as “teleological interpretation of a DO-behavior”. This tells us function of a device is determined contextdependently, though DO-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.

5.1 Several categories of functional concepts

The ontology of the functional concepts is designed to provide a rich and comprehensive vocabulary for both human designers and design supporting systems. It consists of the four spaces as shown in Figure 2. We identified three categories of function: (1) Base-function, (2) function type and (3) meta-function.

5.1.1 Base-function

We call the function defined above a *base-function* of a device, since it is something to do with the entities in the real world. A base-function can be interpreted from object-related aspect (called *object functions*), energy-related aspect (called *energy functions*) or information-related aspect (called *information functions*). For example, the object function and the energy function of a turbine are "to rotate the shaft" and "to generate kinetic energy", respectively.

Figure 2a shows the energy functions organized in an is-a hierarchy with clues of classification. A base-function is defined by minimal conditions about the behavior and the information for its interpretation called Functional Toppings (FTs) of the functional modeling language FBRL (abbreviation of a Function and Behavior Representation Language) [5]. There are three types of the functional toppings; (1) O-Focus representing focus on attributes of objects, (2) P-Focus representing focus on ports (interaction to neighboring devices), and (3) Necessity of objects. For example, a base-function "to take energy" is defined as "an energy flow between two mediums" (a behavioral condition), and "focus on the source medium of the transfer" (functional toppings). The definition of "to remove" is that of "to take" plus "the heat is unnecessary". Thus, "to take" is a super concept

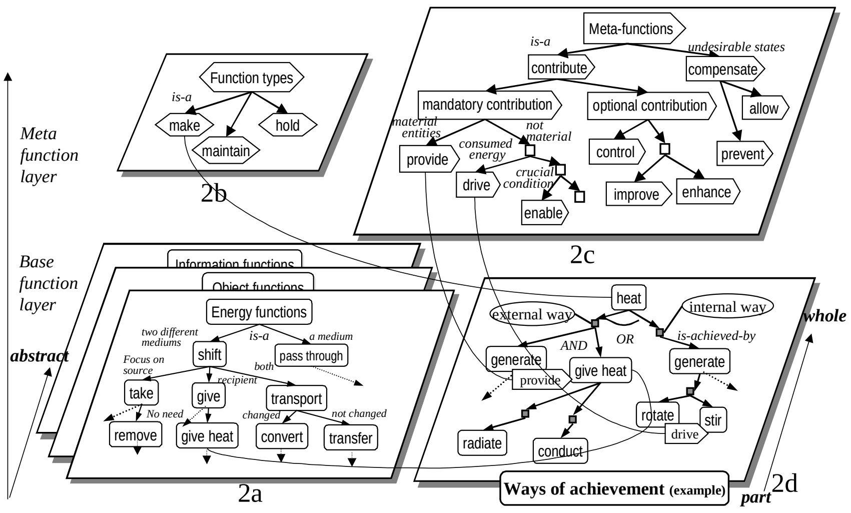

Figure 2: Four kinds of functional planes.

of “to remove” as shown in Figure 2a. The values of O-Focus and P-Focus represent intentional focus of the function. Such a parameter that is a kind of O-Focus and is an attribute of the entity in the focused port indicated by P-Focus is called as the focused parameter. The entity (object or energy) having the focused parameter is called as the focused entity.

The object functions are categorized into the following two categories according to the kinds of the focused parameter. The one is called the *amount function* that changes the parameter values (target objects) or changes the type of the objects, say, from liquid to gas. The other is called the *attribute function* that changes the other attributes of the objects. These categories are used in the definitions of the meta-functions.

Note that definition of base-function using FTs is highly independent of its *realization*, that is, the details of behavior and internal structure of the device. For example, P-Focus specifies not concrete location but abstract interaction with the neighboring devices. The realization is represented by the ways of achievement shown in Figure 2d.

5.1.2 Function types

The function types represent the types of goal achieved by the function [7]. We have redefined the function type as “ToMake”, “ToMaintain”, and “ToHold” as shown in Figure 2b. For example, consider two devices, an air-conditioner (as a heating device) and a heater, having the same function “to give heat”. The former keeps the room temperature at the goal temperature. The latter does not. These are said to be “ToMaintain” and “ToMake”, respectively.

5.1.3 Meta-function

The meta-functions (denoted by *mf*) represent a role of a base-function called an *agent function* (f_a) for another base-function called a *target function* (f_t). A meta-

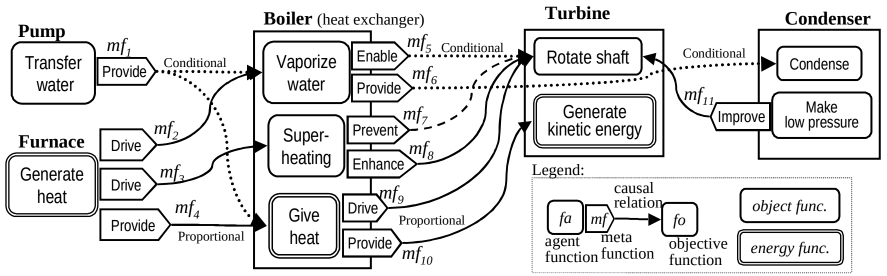

Figure 3: Meta-functions in a power plant (part)

function is not concerned with changes of objects of these devices but with interdependence between base-functions, while other three kinds of functional concepts are concerned with existence or changes of objects.

We have defined the eight types of meta-functions as shown in Figure2c (an is-a hierarchy). Figure 3 shows examples of the meta-functions {mf_1 …mf_{11}} in a simple model of a power plant. Note that furnace which is a sub-component of the boiler is separated from the boiler as a heat exchanger for explanation. The details of definitions and examples in the different domains are shown in [8].

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 ToEnable meta-function for f_t, the authors define ToEnable to have a narrower meaning by excluding the cases of ToProvide and ToDrive as follows:

ToProvide: When a function f_a generates (or transfers) the *materials* which another function f_t intentionally processes, the function f_a is said to perform a meta-function “to provide (material)” for f_t. The material of f_t can be basically defined as input objects (or energy) that will be a part of the output objects (*products*) on which the function f_t focuses. For example, the “to transfer water” function of the pump has a ToProvide meta-function for the “to vaporize” function of the boiler (see mf_1 in Figure 3). If f_t is an attribute function, the function changing the same attribute of the focused material object is said to have a ToProvide meta-function.

ToDrive: When a function generates or transfers such an energy (called *driving energy*) that is *intentionally* consumed by another function f_t, the function is said to have the meta-function “to drive f_t”. For example, the “to give heat” function of the boiler has a ToDrive meta-function for the “to rotate” function of the turbine (see mf_9), because the heat energy is not material of the shaft and is consumed by the rotation. A part of the heat energy is transformed into the kinetic energy that is carried by the focused object (the shaft). On the other hand, for “to generate kinetic energy”, the same function performs not ToDrive but ToProvide (see mf_{10}), because the heat energy is material of the kinetic energy.

ToEnable: This meta-function is used for representing a mandatory condition playing a crucial role in f_t except ToProvide and ToDrive. What we mean by this weak definition is that the conditions such as the existence of the material and the existence of the driving energy are too obvious to be said to enable the function. For example, because the steam of which phase is gas plays a crucial role in occurrence of the heat-expansion process in the turbine and the phase is neither material of rotation nor the consumed energy, the “to vaporize” function of the boiler is said to have a meta-function ToEnable (see *mf_5*).

ToAllow and ToPrevent: These two meta-functions are concerned with the undesirable side effects of functions. A function f_a having positive effects on the side effect of a function f_{t1} is said to have a meta-function “to allow the side-effects of f_{t1}”. If a serious trouble (e.g., a fault) 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}”. For example, the “super-heat” function of the boiler prevents malfunction of the turbine (mf_7), because the steam of low temperature would damage the turbine blade.

ToImprove and ToEnhance: These meta-functions represent *optional* contribution for f_t. The discrimination between ToImprove and ToEnhance is made by increment of the amount of the input energy. For example, the “to keep low pressure” function of the condenser contributes to the efficiency of the “to rotate” function (see mf_{11}) without increment of input energy. That is, it improves the efficiency of “to rotate” function. As to “ToEnhance” meta-function, on the other hand, the “to super-heat” function of the boiler optionally increases the amount of the input energy (mf_8) to increase the output energy, which is enhancement.

ToControl: When a function f_a regulates the behavior of f_t, its meta-function is said to be “to control f_t”. For example, consider a control valve that changes the amount of flow of the combustion gas for the boiler in order to maintain the amount of flow of the steam. It is said to have a meta-function ToControl for the “to vaporize” function of the boiler (not shown in Figure 3).

5.2 Ways of functional achievement

A base-function f_u can be achieved by different groups of sub-functions. We call a group of sub-functions {f_1, ... f_n} constrained by the relations among them (such as meta-functions) *a functional method* of an achievement of f_u. On the other hand, we call the basis of the method *a functional way*. The way is the result of conceptualization of the physical law, the intended phenomena, the feature of physical structure, or components used which explain how the functional methods make sense.

Figure 2d shows some ways of achievement of “to heat an object”, which are described in terms of concepts in other three spaces. There are two ways, that is, the external and internal ways. According to the external way, it is decomposed into two sub-functions, that is, “to generate heat” and “to give heat”. The former should perform a ToProvide meta-function for the latter. In the figure, the latter function can be decomposed according to “radiation way” or “conduction way”.

Note that Figure 2d shows *is-achieved-by* (whole-part) relations among the functional concepts in OR relationship, while Figure 2a shows *is-a* relations as the definitions of them, which are independent of “how to realize them”.

6. Aplications to Industrial Engineering

We now can go into the detailed discussion on systematization of manufacturing process knowledge on the basis of the ontological consideration on function made thus far. Redesign of a manufacturing process through its functional understanding

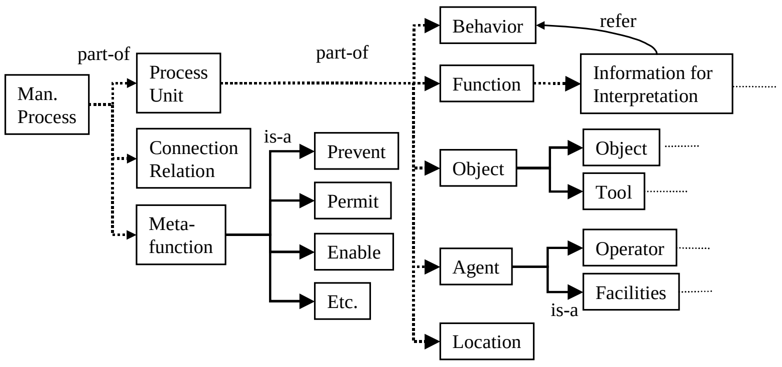

Figure 4: An ontology of manufacturing process (portion).

in industrial engineering is taken as an example task to show how to use the ontologies as well as the way of knowledge systematization of manufacturing processes. A sample ontology of a manufacturing process is also built in terms of functional ontology from device ontology point of view.

6.1 Knowledge model for manufacturing process in industrial engineering

In spite of the long history of the research of industrial engineering, knowledge is not satisfactorily systematized yet and well-established common vocabulary is not available. Furthermore, know-how of redesign of the manufacturing process is kept as heuristics that can be transferred only by apprenticeship. We first tried to establish a system of basic concepts as shown in Figure 4.

Solid lines represent *is-a* relations and dotted lines *part-of* relations. Process unit is a key concept in a manufacturing process and is composed of five sub-concepts such as *Behavior, Function, Agent, Object* and *Location*. An *Agent* processes the *Object* at *Location* by showing *Behavior* that is interpreted as *Function*.

A manufacturing process can be viewed as a sequence of process units (*P*). The model of a process unit consists of *Behavior, Function, Objects* (*O*), *Agents* engaged in the process (*A*) and *Location* where the processing is done. Following our definition of behavior and function, the *Behavior* of a process unit represents the change of the object input to it and is does not change according to the context in any manufacturing processes. Contrary to this, *Function* of the *Behavior* is an interpretation of the *Behavior* under the given context. For example, let us consider a part of a manufacturing process shown in Figure 5. The behavior of the process unit P_2 is "to *assort* input objects O_1 into two categories O_2 and O_3 according to the criteria". Because the next process unit P_4 discharges the assorted objects O_3, we can interpret the behavior of P_2 as the function that "to *remove* the needless objects O_3". Moreover, if the criteria are the quality of the product, it can be called as "*inspect*". This functional interpretation of P_2 depends on the function of P_4. Therefore, under other contexts, P_2 could achieve different functions such as "*select*". Such definition of function is different from others such as that in [9]. Using the functional representation language FBRL [5], we can describe such a functional model as behavior plus functional toppings (FTs). FTs represent the

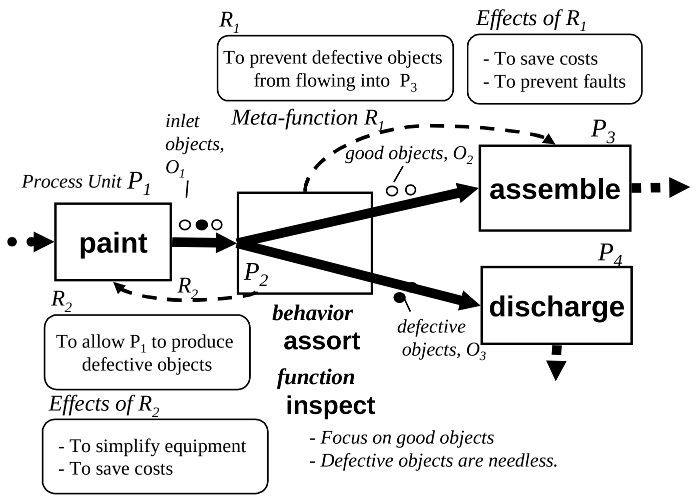

Figure 5: Example of behavior, function, and meta-function of a process

ways of interpretation as discussed in 5.1.1. A model of the "inspect" function mentioned above consists of a behavioral model (constraints over attributes of objects) and two FTs "focus on good objects O_2" and "defective objects O_3 are needless". More detail of modeling based on FBRL is shown in [5].

Connection relation represents the structure of the manufacturing process and meta-function represents how each process unit contributes to each other. A process unit contributes to other process units with specific goals.

Figure 5 also shows the meta-function of the inspection process P_2 mentioned above. It *allows* the paint process P_1 to produce the defective objects, and *prevents* them from flowing into the assembling process P_3. The former contributes to simplifying the equipment of P_1 and hence to saving of costs. The later prevents faults of P_3 and contributes to saving costs.

Such meta-functions represent the justification of the existence of the process units. Justification of the inspection process P_2 is production of the defective objects by the paint process. Such justification is a part of "design rationale" of the manufacturing process.

Figure 6 shows is-a hierarchy of "Process unit". A process unit has sub-concepts such as *Operation, Delay, Transportation and Inspection.* One of the contributions of our systematization of manufacturing process knowledge is the explication of attributes on which these four activities operate. We identified four categories of attributes such as *physical attributes, information attribute, spatial attribute and temporal attribute*. On the basis of these categories of attributes, we define the above four activities together with *Storage* activity as follows:

- *Operation*: change of physical attributes
- *Transportation*: change of spatial (location) attributes
- *Inspection*: change of information attributes
- *Delay*: change of temporal attributes (the change can be done without permission)
- *Storage*: change of temporal attributes (the change cannot be done without permission)

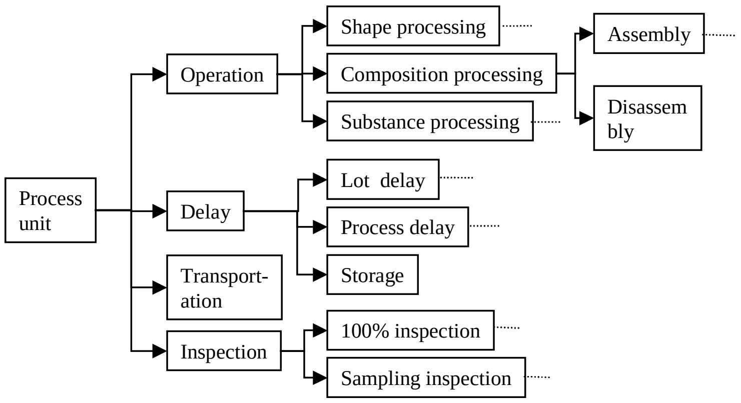

Figure 6: Is-a hierarchy of process unit.

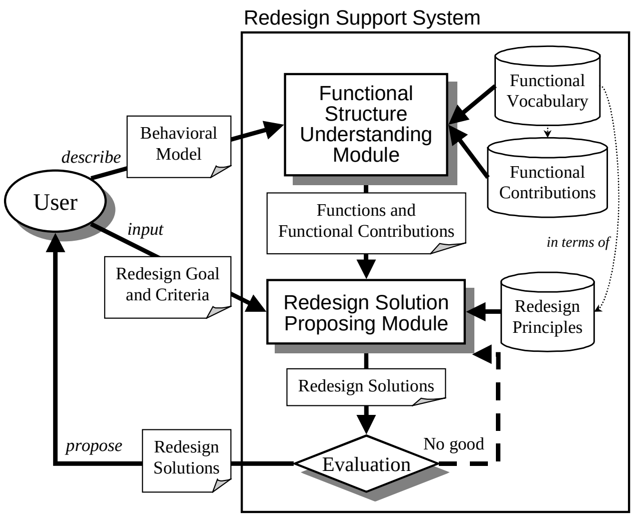

Figure 7: Framework of the redesign support system

6.2 Redesign of a manufacturing process

Figure 7 shows the framework of the redesign support system we are developing. It consists of the following two major functional modules:
(1) Functional structure understanding module and
(2) Redesign solution proposing module.

6.2.1 Functional understanding

The former module can generate plausible functional structures from the given behavior models of the target process. Firstly, the module generates all possible functional interpretations of the given behavior according to a functional vocabulary. It contains definitions of a number of functional concepts as constraints over FTs. Since the search space over FTs is limited, the functional

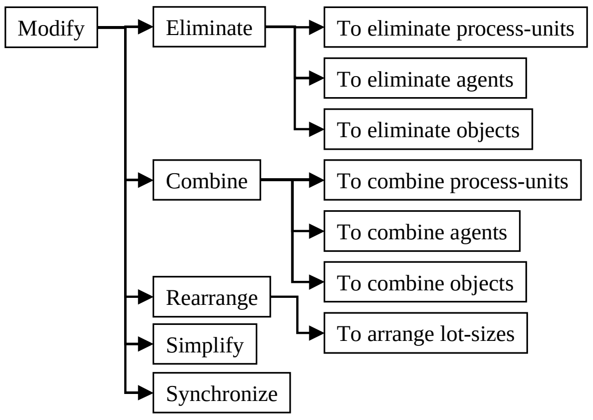

Figure 8: Is-a hierarchy of Modify operation.

Table 1: Examples of the redesign principles

	Omission Principle	**Agent Sharing Principle**
Goals	Reduction of processing time and/or costs	Reduction of costs for agents
Targets	A process unit P_1	Process units P_1,P_2 , Agents A_1,A_2
Conditions	P_1 has no functional contribution	The performance of A_1 is enough to do P_1,P_2
Changes	To omit P_1	To remove A_2 and to engage A_1 in P_2
Effects	Reduction of processes	Reduction of agents

understanding module generates all tuples of values of FTs and then matches them with definitions in the functional vocabulary.

Next, meta-functions among the generated functions are identified according to the definitions of meta-functions shown in Section 5.1.3. The algorithm is shown in [8]. It enables the module to identify plausible functional interpretations of *P* from the generated possible functional interpretations, because a functional interpretation that does not contribute to any processes is not plausible. Such contributions cannot be captured by causal relations as pointed out in [9].

6.2.2 Redesign

The later module proposes how to change the manufacturing process (redesign solutions). Then, they are evaluated according to the given criteria. The module uses the redesign principles knowledge representing general improving principles in Industrial Engineering (IE), called ECRS principles shown in Figure 8. They are highly abstracted in terms of the functional vocabulary and hence reusable. Table 1 shows examples of them.

6.3 Example of redesign

Figure 9 shows an example of redesign of a manufacturing system. Firstly, the redesign system identifies the meta-functions (R_1 and R_2) shown in Figure 9a from

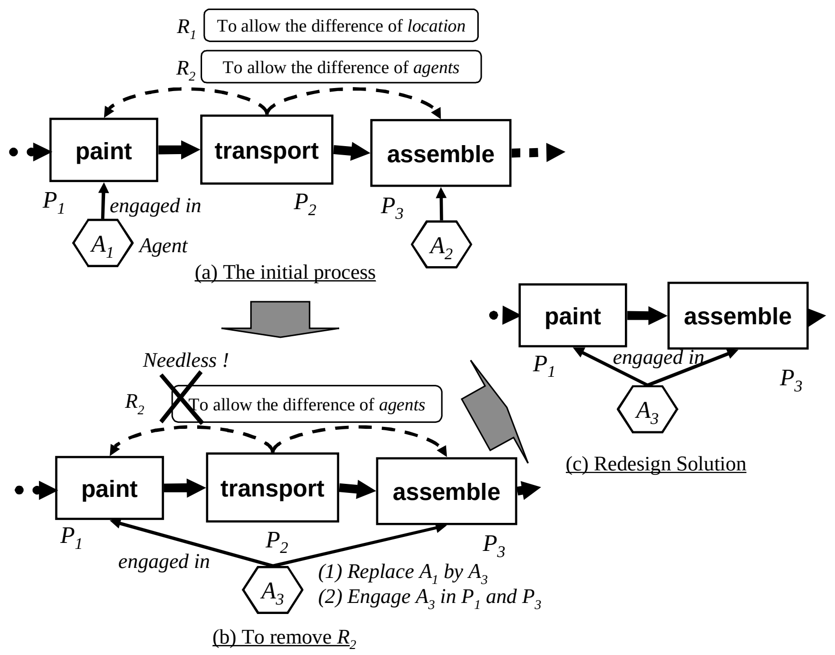

Figure 9: Example of redesign

the given behavioral model. The transportation process unit P_2 allows the two differences between P_1 and P_3, that is, the difference of location (R_1) and the difference of agents (R_2). In other words, the two differences are the justification for the existence of P_2.

Next, the system tries to generate redesign solutions. The goal (criteria) of redesign here is reduction of the processing time. Thus, the system tries to omit P_2 according to the "omission principle" shown in Table 1. In order to satisfy its condition, that is, no meta-function, the redesign system has to remove the meta-functions by changing P_1 and/or P_3.

The meta-function R_1, that is, to allow the difference of location, can be removed by changing the layout. On the other hand, for the meta-function R_2, that is, to allow the difference of agents, the system tries to apply the "agent sharing principle" shown in Table 1. In the cases where the performances of the agent A_1, A_2 are not enough to do so, the system also applies the "agent replacement principle". Then the two processes share an agent A_3 (Figure 9b). At this point, there is no meta-function of P_2 . Then, the process unit can be omitted (Figure 9c).

Lastly, the redesign solution is evaluated. Its effects include saving of the processing time by omitting the transportation and reduction of costs for an agent. The side effect is costs for replacing the agents.

Another example is shown in Fig. 10. The story is as follows: The quality of painting was turned out not to be satisfactory. Diagnosis revealed the fact that the power of compressor used for painting is not enough. The compressor was shared by shot blast that gets rid of the stains on the painting objects, so the possible solutions were

(1) to replace the compressor with that of more power

(2) to install another compressor for the painting unit.

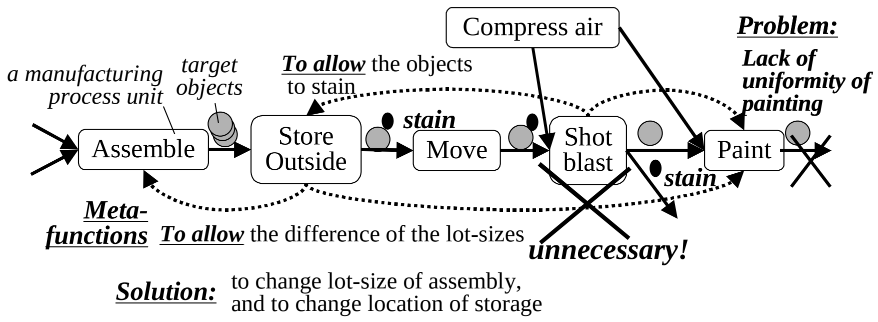
Compress air
Problem:
Lack of uniformity of painting
a manufacturing process unit
target objects
To allow the objects to stain
stain
Assemble
Store Outside
Move
Shot blast
Paint
stain
Meta-functions
To allow the difference of the lot-sizes
unnecessary!
Solution: to change lot-size of assembly, and to change location of storage

Figure 10: Another example (Shot blast elimination case).

Needless to say, these solutions are not good because it costs a lot. A better solution has been obtained as follows: Investigation on meta-functions of the shot blast uncovered that it allows the objects to stain and to prevent from low quality of painting and that of storage of objects outside are to allow difference of the lot-sizes of Assembling unit and Painting unit. Considering the fact that getting objects stained is caused by the storage outside, if we find a space for storing objects inside, we can prevent objects from being stained, and hence we can eliminate the shot blasting unit, and hence the painting unit can enjoy necessary power of compressor by using the current one to attain the required quality of painting.

7. Related Work

7.1 Ontology of functional concepts:

The functions in [3,5,6,10,11] represent abstract behavior of components and are defined as base-functions in our framework. The meta-function represents a role for another function without mention of changes of incoming or outgoing objects of components and hence is totally different from such base functions. In [12,13], function is defined as a kind of hierarchical abstraction of behavior. It corresponds to the is-achieved-by relations among functions in our framework.

The CPD in CFRL [13] represents causal relations among functions. Lind categorizes such relations into Connection, Condition and Achieve [10]. Rieger identifies “enablement” as a type of the causal relation between states and action [14]. The meta-functions are results of interpretation of such causal relations between functions under the role of the agent function for the target functions.

In [15], the sets of “primitives of behavior” are proposed. Lind identifies a few general functions which are categorized into multiple levels [11]. We added more intention-rich concepts and organized in is-a and part-of hierarchy. Furthermore, we identify some types of the meta-functions.

In Value Engineering [16], standard sets of verbs (i.e., functional concepts) for value analysis of artifacts are proposed [17]. It enables the human designers to share descriptions of functions of the target artifacts. However, they are designed only for humans, and there is no machine understandable definition of concepts.

In a literature on design, many general “patterns” of synthesis are proposed (e.g., [18]). Our ways of achievement, however, explicitly represent the feature of achievement such as theory and phenomena. The importance of such key concepts in design is pointed out in [19]. They enable the redesign system to facilitate the smooth interaction between models at the structural and functional levels.

7.2 Ontology-based approach for knowledge modelling

Borst et al. [20] discuss very similar idea of layered ontologies for engineering composed of Mereology, Topology, System theory, Component ontology, Process ontology and EngMath ontology. While their ontologies are well-formalized, they do not discuss the function or behavior which are our main topic. Although their Component ontology should correspond to our device ontology, it mainly is concerned about structural aspect and leaves content part for Process ontology. Our device ontology is self-contained for modeling artifacts.

8. Conclusions

We have discussed ontology-based knowledge systematization. Functional knowledge is set to the main target for investigation, since **function**, which has left untouched to date, plays the key role in AI-based knowledge modelling. Ontological engineering contributes to our enterprise in some respects:

1. It bridges the gap between humans and computers by providing well-structured vocabulary (concepts) in terms of which we can model functional knowledge,
2. it gives a system of concepts used as foundation of knowledge modeling,
3. and hence, it enables us to avoid ad hoc knowledge representation.

We applied the results of ontological engineering of functional knowledge to modelling of manufacturing processes and demonstrated how the redesign system works using systematized knowledge. Future plans include incorporating Value Engineering (VE) results [16] into the redesign module to realize creative redesign as well as implementation of the system.

Acknowledgements
This research has been partly supported by the Japan Society for the Promotion of Science (JSPS-RFTF97P00701) and GNOSIS, IMS. The authors are grateful to Dr. Fuse, Sumitomo Electric Industries for his help on manufacturing process modelling.

References

[1] Mizoguchi, R., and Ikeda, M. 1997, Towards ontology engineering. In *Proc. of PACES/SPICIS '97,* 259-266.
[2] Mizoguchi, R. et al., 1999, A Methodology of Collaborative Synthesis by Artificial Intelligence, 2nd International Workshop on Strategic Knowledge and Concept Formation, pp.221-232.

[3] Chandrasekaran, B., and 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*.

[4] Iwasaki, Y., et al., 1993, How things are intended to work: Capturing functional knowledge in device design. *Proc. of IJCAI-93*, pp.1516-1522.

[5] Sasajima, M.; Kitamura, Y.; Ikeda, M.; and Mizoguchi, R., 1995, FBRL: A Function and Behavior Representation Language. In *Proc. of IJCAI-95*, 1830-1836

[6] Umeda, Y. *et al.*, 1990. Function, behavior, and structure. *AI in Engineering*, 177-193, 1990.

[7] Keuneke, A. M., 1991, Device representation: the significance of functional knowledge. *IEEE Expert*, Vol.24, pp.22-25.

[8] Kitamura, Y., and Mizoguchi, R. 1999. Meta-functions of artifacts, Papers of *13th International Workshop on Qualitative Reasoning (QR-99)*, 136-145

[9] Chandrasekaran, B., Goel, A. K., and Iwasaki, Y., 1993, Functional representation as design rationale. *COMPUTER*, pp.48-56.

[10] de Kleer, J. 1984, How circuits work, *Artificial Intelligence* 24:205-280.

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

[12] Sembugamoorthy, V., and Chandrasekaran, B. 1986, Functional representation of devices and compilation of diagnostic problem-solving systems. In *Experience, memory and Reasoning*, 47-73.

[13] Vescovi, M., et al., 1993, CFRL: A language for specifying the causal functionality of engineered devices. In *Proc. of AAAI-93,* 626-633.

[14] Rieger, C., and Grinberg, M., 1977, Declarative representation and procedural simulation of causality in physical mechanisms. In *Proc. of IJCAI-77*, 250-256.

[15] Hodges, J., 1992, Naive mechanics - a computational model of device use and function in design improvisation. *IEEE Expert* 7(1):14-27

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

[17] Tejima, N. *et al.* eds, 1981, Selection of functional terms and categorization. Report 49, Soc. of Japanese Value Engineering (In Japanese).

[18] Bradshaw, J. A., and Young, R. M., 1991, Evaluating design using knowledge of purpose and knowledge of structure. *IEEE Expert* 6(2):33-40.

[19] Takeda, H., et al., 1990, Modeling design processes. *AI Magazine*, 11(4), 37-48.

[20] Borst P., Akkermans, H., and Top, J., 1997, Engineering ontologies, Int. J. Human-Computer Studies, 46, pp.365-406.