Functional Ontology for Functional Understanding

Yoshinobu Kitamura and Riichiro Mizoguchi

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

Abstract

This article discusses an ontology of generic functional concepts of artifacts, called a *functional ontology*. We discuss the requirements for the ontology and the characteristics of some existing functional modeling languages. We present a functional ontology described by our functional modeling language FBRL (Sasajima et al. 1995) in terms of its mapping primitives between behavior and function. They make the definitions independent of their implementation.
The functional ontology enables us to realize *a functional understanding system* which identifies functional structures of an artifact from its behavioral and structural model. It plays a crucial role to specify and limit the functional space. This article also presents a framework of the functional understanding system and an example of functional understanding of a power plant.

Introduction

Recently, much attention has been paid on *ontology* aiming at a basis for modeling knowledge. It is an explicit specification of conceptualization (Gruber 1993) and provides primitive vocabulary for knowledge-based systems (Mizoguchi and Ikeda 1997). The importance of explicit conceptualization for reusability of knowledge has been widely recognized (Abu-Hanna and Jansweijer 1994, Gruber 1993, Mars 1995, Mizoguchi and Ikeda 1997).

This research is an attempt to establish a functional ontology which consists of functional concepts representing function of artifacts. In general, an ontology should satisfy the following requirements:

- Sophisticated articulation of the target object or world
- Explicit definition of concepts and relations among them
- Generality and Comprehensiveness

Therefore, our goal here is to identify a finite number of meaningful and generic concepts representing functions of artifacts and then define each concept explicitly.

A lot of research has been carried out on functional representation of artifacts (Chandrasekaran and Josephson 1996, de Kleer 1984, Lind 1994, Price and Pugh 1996, Sasajima et al. 1995, Sembugamoorthy and Chandrasekaran 1986, Umeda et al. 1990, Vescovi et al. 1993). Almost all of functional models in the conventional research, however, are described by human modelers, and are specific to the target system. Thus, there are only a few generic functional concepts. De Kleer identifies a few possible functions of some components in electronic circuits such as "I-LOAD" of a resistor (de Kleer 1984). Lind identifies a few general functions such as "storage of energy" (Lind 1994).

On the other hand, in Value Engineering research (Miles 1961), standard sets of verbs (i.e., functional concepts) for value analysis of artifacts are proposed (Tejima et al. 1981). 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.

We have proposed a functional modeling language named FBRL (Sasajima et al. 1995), which enables us to build such a functional ontology. FBRL introduces *functional toppings* (FTs) which are a set of primitives for mapping from behavioral space to functional space. As shown in this article, they enable us to define generic functional concepts (see Figure 4). For example, the definition of the function "give heat" consists of "shifting heat energy" (a behavioral condition) and "focus on the medium receiving the heat" (a functional topping).

The functional ontology provides a conceptual vocabulary in the functional space, which plays a crucial role to specify and limit the functional space. It contributes to the following kinds of problem solving:

- Functional Understanding by enabling mapping from behavioral space to functional space.
- Design (redesign) by enabling to understand the requirements and operate them.

The functional understanding is to identify functional structures of an artifact from its behavioral and structural model. Although it is in principle difficult because the search space of function is huge, the functional ontology plays a role to limit the search space. It provides such primitives in the functional space that are targets in the mapping, and screens out meaningless functional interpretations.

Moreover, the functional ontology enables the design problem solvers to understand the requirements and to reason in the functional space which is compatible with the terms used in them. The functional understanding is a

subtask of the redesign task (Goel and Chandrasekaran, 1989) which is a subtype of the design tasks.

In this article, we present a functional ontology based on FBRL together with a framework of functional understanding based on the ontology. First, we discuss the requirements for a functional ontology and relationship between behavior and function in Section 2. On the basis of the discussion, the framework of a functional ontology and definitions of behavior and function are shown. We outline the FBRL modeling in Section 3, and present a functional ontology based on FBRL in Section 4. Next, the types of functional relations are shown in Section 5. Then, the steps of the functional understanding task are discussed in Section 6 to 9 together with an example. Section 10 discusses related work.

What is a Functional Ontology?

In this section, we discuss requirements for a functional ontology and then characteristics of behavior and function to be captured by a modeling language through comparison of some conventional functional modeling languages.

Requirements

A functional ontology is an ontology of functional concepts in a specific target domain. According to the general requirements for ontologies mentioned in the introduction, the requirements for a functional ontology is obtained as specification of them as follows:

- Comprehensive articulation of generic function
- Definition in terms of mapping primitives which help mapping from behavioral space to functional space.
- Independence of implementation

First, in order to specify the functional space, ideally, all functional concepts recognized by humans in the target domain should be defined.

Secondly, because the search space of functional concepts is huge, it is a crucial issue that how we limit the mapping relations between behavior and function to a reasonable size. We argue that such *mapping primitives* that can specify the various ways of interpretations of a behavior play the role. They allow the understanding system to generate a limited number of candidates of functional concepts in the functional space from a given component model. Thus, functions should be defined in terms of them.

Lastly, it is desirable that a definition of a functional concept is independent of its implementations, that is, how to realize it, in order to make the ontology general. The implementation of a function should be viewed as a composition of (1)behavior and structure dependent part and (2)behavior structure-independent decomposition of a function into sub-functions. To maximize the freedom of functional interpretation, the dependency should be minimized.

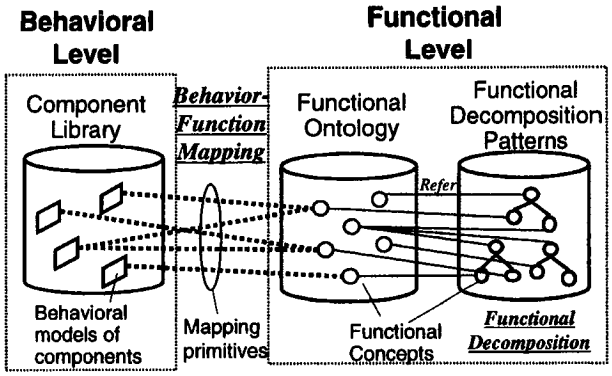


Figure 1: A conceptual framework of a functional ontology

Conceptual Framework

On the basis of the discussion above, we have came up with a conceptual framework of the functional ontology shown in Figure 1 where the functional level is detached from the behavioral level. The relationship between the two levels is maintained by the mapping primitives mentioned above, which realizes maximization of its independence of behavior and structure. A functional concept is defined independently of functional decomposition, which is done using *functional decomposition patterns*. Because they are described in terms of the functional concepts, the patterns are generic and independent of behavior and structure.

On the basis of the functional ontology, a concrete target system will be functionally understood as shown in Figure 2 where the behavior-function mapping is identified as a separate operation from the hierarchical operation (i.e., decomposing and understanding operations) at the functional level. The horizontal *behavior-function* axis represents *role* intended by the designer and *interpretation* by the understanding system. The operation of mapping from behavior to function is done according to the mapping primitives and the functional concepts in terms of them.

The vertical *hierarchical* axis represents *grain sizes* of representation and the decomposition that how a super-function is achieved by sub-functions. It explains "how it contributes to the whole system". The functional hierarchy is understood according the generic functional decomposition patterns.

Lind shares the importance of the detachment of behavior-function relation from functional decomposition with us (Lind 1994). Lind calls these relations "means-ends" and "whole-part", respectively.

In contrast, in FR (Iwasaki and Chandrasekaran 1992, Sembugamoorthy and Chandrasekaran 1986) and CFRL (Vescovi et al. 1993), function is defined as a kind of hierarchical abstraction of behavior. Their functional models are described as sequences of partial states of the behavior, so that they depend on their implementations.

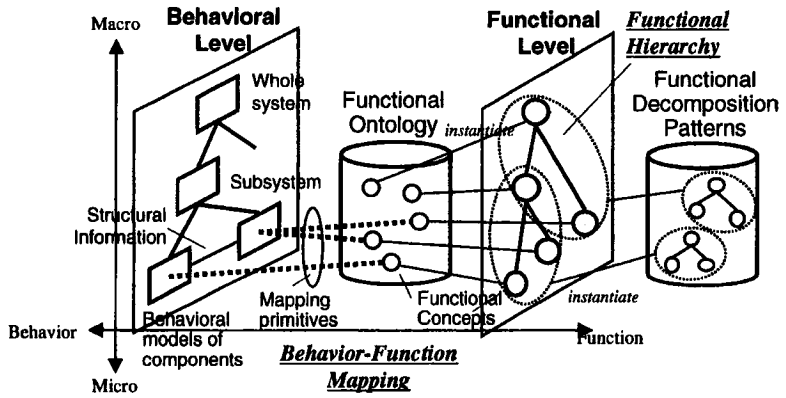


Figure 2: Result of understanding of a target system based on the framework

Recently, Chandrasekaran points out the importance of implementation-independent functional models and proposes a representation of functions as effect (Chandrasekaran and Josephson 1996).

Definitions

In the following paragraphs, we define some important concepts and discuss characteristics of them.

Behavior. The behavior can be defined as temporal changes of parameter values. The model of a target system for generating the behavior consists of the behavioral models of components and structural information representing connections among them. The behavioral model of a component is ideally independent from the context which the component is used in (the no-function-in-structure principle (de Kleer and Brown 1984)). For example, the behavior of a heat exchanger can be described by the constraints over parameters such as temperatures of the mediums and the heat resistance. When it is used as a heater or a radiator, it achieves different functions ("give heat" or "remove heat", respectively). The behavior is, however, the same. Of course, no behavior model is completely independent of the context. Our framework aims at identifying the plausible functional contexts from the given behavior model.

Function. A function is defined as a result of interpretation of a behavior under an intended goal (Sasajima et al. 1995). Although, the functional models in (Chandrasekaran and Josephson 1996, Price and Pugh 1996, Umeda et al. 1990) are described as partial states associated with the goal, a functional model is not composed of only states. For example, the discrimination between the function of a heat exchanger as a heater "give heat" and the function of that as a radiator "remove heat" requires the information whether the heat is needed for the system or not. Keuneke also discusses types of functions (Kueneke 1991). It is needed to identify such mapping primitives between behavior and function that enable the reasonable and effective interpretation of the behavior.

Functional Concept. The functional concepts are conceptual classes of concrete functional models (instances). They are defined in the functional ontology in an implementation-independent manner. They are used as a conceptual vocabulary in the functional space to limit the reasoning space and to describe functional knowledge such as the functional decomposition patterns.

Functional Relation. The functional relations are relations among functions in the same grain size. The typical one is the causal relation among functions. Moreover, we believe that some of the functional types discussed by Keuneke (Kueneke 1991) (e.g., ToPrevent) represent characteristics not of function but of functional relations among two functions. The categories of them should be identified.

Functional Hierarchy. The functional hierarchies represent *functional-part-of* relations among functions in different grain sizes. Although they in many cases correspond to the structural hierarchies (i.e., system-subsystem-component relations) at the behavioral level as discussed in (Snooke and Price 1997), it is not always the case. The "structure" of target systems is one of the *viewpoints* for understanding functional hierarchies. A target system can be functionally understood as some different functional hierarchies according to the different viewpoints.

Overview of FBRL

This section outlines a functional modeling language named FBRL (abbreviation of Function and Behavior Representation Language) (Sasajima et al. 1995). An FBRL model of a target system consists of component models and structural information among component. An FBRL component model consists of a behavioral model and a set of the mapping primitives called Functional Toppings (FTs) as shown in Figure 3.

The behavioral component model consists of inflow objects, outflow objects, connection ports, and constraints over parameters.

There are four types of the functional toppings; (1)O-Focus representing focus on attributes of objects, (2)P-Focus representing focus on ports (interaction to neighboring components), (3)FuncType representing types of contribution (extended one of those defined by Keuneke (Keuneke 1991), and (4)Necessity of objects. For more details of FBRL, see (Sasajima et al. 1995).

Note that such FTs of a function are highly independent of its implementation, that is, details of behavior and internal structure of the component. For example, P-Focus specifies not concrete location but abstracted interaction with the neighboring components.

Functional Ontology based on FBRL

The functional ontology consists of functional concepts organized in an *is-a* hierarchy with clews of classification shown in Figure 4. The definition of a functional concept consists of a label representing the concept and conditions of behavior and functional toppings. For example, the functional concept "take energy" is defined as "an energy flow between two mediums" (a behavioral condition), and "focus on the medium transferring the heat" (a functional condition) as shown in Figure 5. Moreover, the definition of "remove" is that of "take" plus "the heat is unnecessary". Thus, "take" is a general (super) concept of "remove" as shown in Figure 4.

The functional ontology provides conceptual vocabulary for describing functional knowledge such as the functional decomposition patterns described latter. Because the ontology makes the knowledge detached from behavior and structure, the knowledge is reusable.

Currently, the functional ontology is designed for fluid-related plants such as power plants and chemical plants. It includes such functional concepts that represent changes of energy and those of fluid which carries the energy. It does not cover mechanical phenomena.

It has been successfully applied to a simple model of a power plant shown in this article and a concrete chemical plant. The functional models of the plants share many functional concepts except those specific to the chemical domain such as "react".

Types of Functional Relations

The functional relations are relations among functions in the same grain size. These relations can be categorized into some types, which provide the vocabulary for describing constraints over relations among sub-functions in the functional decomposition patterns as shown latter.

Firstly, the functional relations can be categorized into two major types, *causal-type* and *structural-type*. The former functional relations are defined by causal relations at the behavioral level. We call a parameter that a function focuses on the *functional parameter* of the function. When there is a causal relation between two functional parameters of two different functions, there is a causal-type

```
Mode1  /* a behavioral mode */
  Precondition:     /* condition for a behavioral mode */
  Behavior:
    Objects:        /* objects input to or output from component */
    SubComponents:  /* names of subcomponents */
    MP-Relations:   /* relations among input materials and output
                       products */
    SameClass:      /* sameness of the class of in/out objects */
    InherentParams: /* parameters inherent in the component */
    Ports:          /* connection with neighbor component */
    QN-Relations:   /*qualitative relations among parameters */
  FT-Set1
    FuncType:       /* type: ToMake, ToMaintain, ToHold */
    O-Focus:        /* focus on a certain class of object */
    Focus:          /* focus on a certain input-output relation */
    Necessity:      /* necessity of output objects */
  FT-Set2 ...
Mode2 ...
```

Figure 3: The Scheme of FBRL Component Model

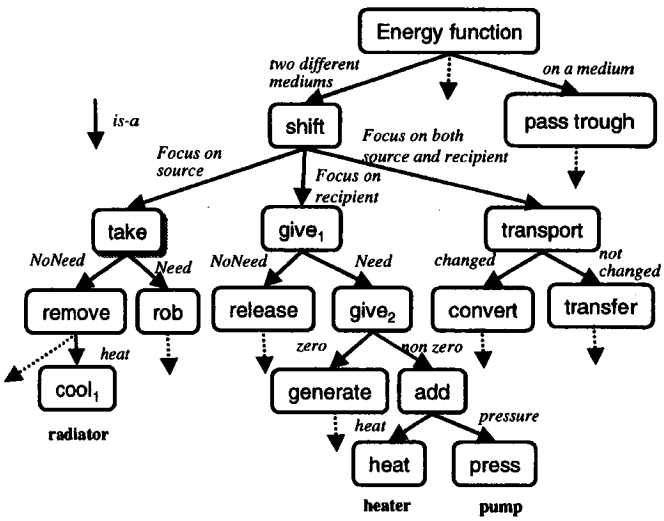


Figure 4: *is-a* hierarchy of functional concepts (part)

```
(define-function-class take-energy (?fd)
   :label ("take energy")
   :subtype-of ('shift-energy)
   :e-def (and (functional-part ?fd ?f)
     (behavioral-part ?fd ?b)
     (has-energy-objects ?b (?e0 ?e1 ?e2))
     (has-medium ?b ?m0 ?e0)
     (has-medium ?b ?m1 ?e1)
     (has-medium ?b ?m2 ?e2)
     (has-ports ?b (?port0) (?port1))
     (exists-in (?m0 ?e0) ?port0)
     (exists-in (?m1 ?e1) ?port1))
   :behavioral-condition (and
     (mp-relation ?e0 ?e1)
     (mp-relation ?e0 ?e2)
     (mp-relation ?m0 ?m1)
     (not (mp-relation ?m0 ?m2)))
   :functional-condition (and
     (focus-on-ports ?f ?port0 ?port1)))
```

Figure 5: A definition of the functional concept "take"

functional relation between these functions. The causal-type relations between two functions *f1* and *f2* can be categorized into the following four sub-types, where *fp1* and *fp2* are functional parameters of *f1* and *f2*, respectively;

- Proportional-type. A positive change of *fp1* causes a positive change of *fp2*. For example, in a power plant shown in Figure 6 (we use it as an example of a target system in this article), the relation between the temperature of the outlet steam of the boiler and the number of revolutions of the shaft of the turbine is proportional.
- Precondition-type. There is a causal relation between *fp1* and the precondition of *f2*. If the precondition is not satisfied, *f2* will not be achieved discretely. For example, the condenser changes the phase of the medium from gas to liquid. The gas phase is a precondition of the condenser.
- Efficiency-type. The optional existence of *f1* causes a positive change of the efficiency of *f2*. There is a causal relation between *fp1* and one of the parameters which represent the efficiency of *f2*. For example, the function "heat insulate" of the turbine contributes the efficiency of the function "expand".
- Preventing-type. If *f1* is not achieved, a serious trouble (e.g., faults) will occur in *f2*. For example, the "super-heat" function of the boiler prevents the fault of the turbine, because the low temperature of the steam would damage the turbine blade.

On the other hand, the structural-type causal relations are defined in terms of structural information. These are categorized into subtypes such as series, parallel, sequential, simultaneous and feedback.

The Steps of Functional Understanding

The functional understanding task is to identify functional structures of an artifact from the given structural information and behavioral models of components of the finest grain size and the qualitative behavior. The behavior is generated by a qualitative reasoning system from the structural and behavioral models.

The three steps of functional understanding task are shown in Figure 7. First, given behavioral component models, all possible functional interpretations of behavior of each component are generated (called behavior-function mapping). Next, the functional relations among generated functional interpretations are identified (called functional relation understanding). Lastly, from the functional interpretations and relations among them, functional hierarchies are generated (called functional hierarchy understanding). Although many candidates of the functional interpretations are generated by the first step, plausible functional interpretations are identified by the second step and the third step.

This understanding procedure is done in a bottom-up manner. If the function (or goal) of the whole target system is given, the understanding system generates the functional

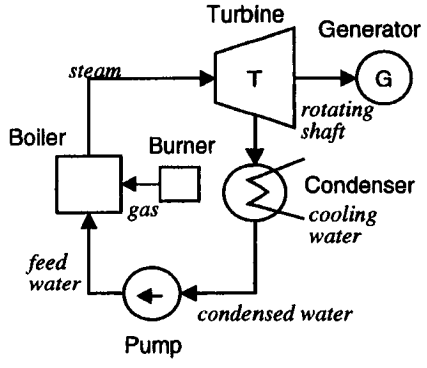


Figure: 6: A power plant

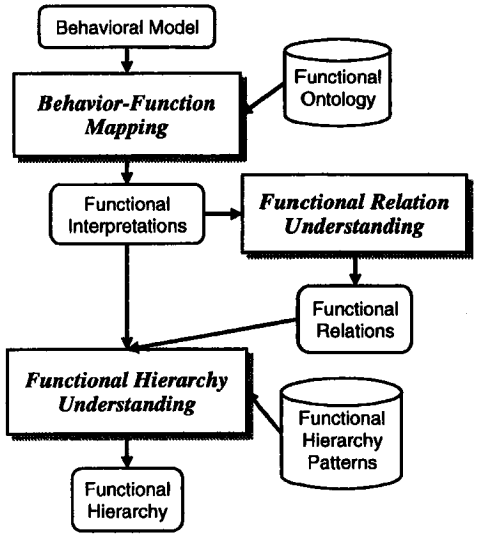


Figure 7: The steps of the functional understanding

hierarchies of which the function is the top-most function (the function of the system as a whole). Otherwise, various top-most functions are generated. However, on the assumption that the whole system achieves a function which has some effects to the outside of the system, the top-most functions should be associated with the parameters of the output from the target system. For example, in the power plant, the top-most functions should be associated with the electricity of the generator, or the temperature of the cooling water output from the condenser. Thus, the top-most functions are "to generate electricity" or "to warm the (cooling) water". Although we do not recognize the latter as the function of the power plant in general, such recognition requires the model including the context which the power plant is used in.

Behavior-function Mapping

The behavior-function mapping step generates all possible functional interpretations of behavior of each component

from the given behavioral component models. The functional ontology enables us to realize the behavior-function mapping. Although it is in principle difficult because the search space of function is huge, the mapping primitives, the FTs in FBRL, play a crucial role to limit the search space.

Firstly, possible candidates can be exhaustively generated as all tuples of possible values of FTs context-independently. Next, the understanding system screens out meaningless ones by matching them with functional concepts. Such functional interpretations that match with no functional concept are screened out as a meaningless interpretation assuming the completeness of the ontology in the functional space.

It should be noted that the generated functions are limited by the given behavioral model. The behavioral model should support the all possible functions.

Example

Figure 8 shows the behavior-function mapping of the boiler. The input shown in the upper part of the figure consists of objects, connection ports, and behavioral constraints such as MP-relations. Firstly, possible values of FTs are generated exhaustively. For example, a functional interpretation f_3 consists of O-Focus on the "phase" parameter and P-Focus on the inlet water and the outlet steam. Then, the functional interpretation is successfully matched with a functional concept "vaporize". In contrast, such many functional interpretations that match with no functional concept (e.g., f_4 in the figure) are screened out as a meaningless interpretation.

Nevertheless, some possible functional interpretations such as "give heat", "expand" and "remove heat" are generated. Neither of the last two concepts can be functions of the boiler. They will be deleted by the next step.

Functional Relation Understanding

The causal-type functional relations among the functional interpretations are identified according to behavioral causal relations among them generated by a qualitative reasoning engine. We use our original one (Kitamura et al. 1996, 1997a, 1997b) which is categorized as a type of the reasoning method proposed by de Kleer and Brown (de Kleer and Brown 1984). Given an abnormal value of a parameter, it can generate causal chains representing the effect of the abnormality.

For each functional interpretation *f1*, a causal chain from the abnormality of the functional parameter *fp1* of *f1* is generated by the qualitative reasoning engine. If the functional parameter *fp2* of a functional interpretation *f2* is found in the generated causal chain, there is a causal-type functional relation between *f1* and *f2*. If it is proportional one, it is a proportional-type. If the precondition of *f2* (described in the "precondition" slot of the behavioral model as shown in Figure 3) is also found in the causal chain, it is called precondition-type. On the other hand, a

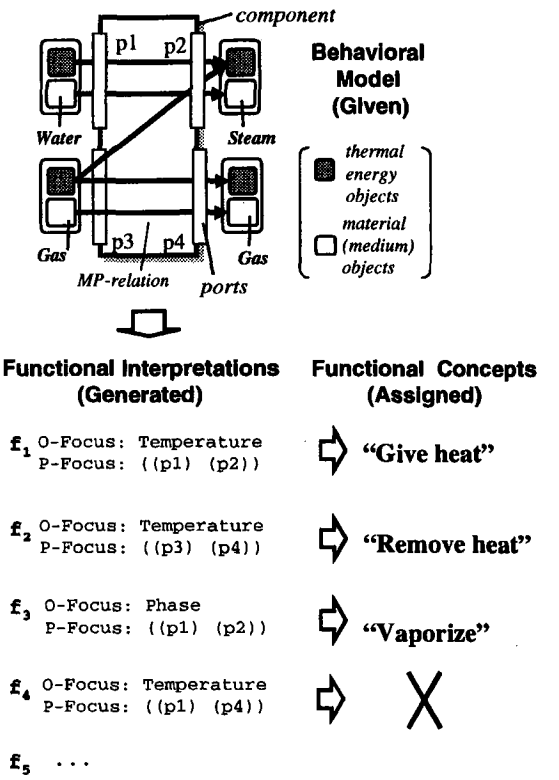


Figure 8: Behavior-function mapping of the boiler

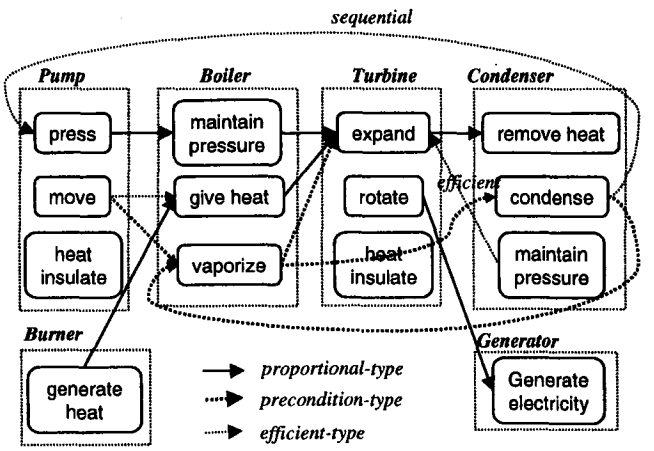


Figure 9: Functional relations of the power plant

part of the structural-type functional relations is identified according to the structural information among components.

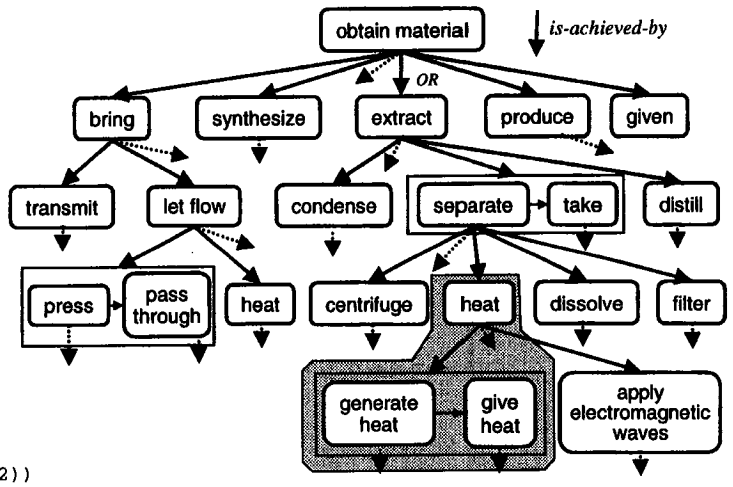
According to the identified functional relations, the understanding system deletes such meaningless functional interpretations that do not contribute to any others.

Example

A part of functional relations identified is shown in Figure 9. It enables the understanding system to delete the

```
(define-functional-decomposition-pattern
         heat-by-giving-heat-energy (?fhd)
 :label ("heat object")
 :decomposition (and
   (has-super-function ?fhd 'heat ?fd0)
   (has-sub-function ?fhd
       'generate-heat-energy ?fd1)
   (has-sub-function ?fhd
       'give-heat-energy ?fd2))
 :e-def (and
   (focus-on-objects ?fd0 ?obj01 ?obj02)
   (focus-on-out-object ?f1 ?obj12)
   (focus-on-objects ?f2 ?obj21 ?obj22)
   ...)
 :behavioral-condition (and
   (obj-connection ?obj12 ?obj2e)
   (is-identical ?obj01 ?obj21)
   (is-identical ?obj02 ?obj22))
 :functional-relation-condition
   (proportional-functional-relation ?fd1 ?fd2))
```

(a) A definition of a functional decomposition pattern "heat-by-giving-heat-energy"



(b) Functional decomposition patterns of "obtain" (part)

Figure 10: Functional Decomposition Patterns

functional interpretations "expand" and "remove heat" of the boiler, because they do not contribute to any others.

Functional Hierarchy Understanding

Functional Decomposition Pattern

A function is achieved by a sequence of sub-functions. A combination of a super-function and its sub-functions is called a functional decomposition pattern. Its definition consists of a super-function, sub-functions, functional relations among sub-functions, and behavioral conditions. These functions are described in terms of the functional concepts. For example, a super-function "heat object" has two sub-functions; "generate heat" and "give heat". There should be a proportional-type functional relation among them. The behavioral condition is that the objects receiving the heat are identical. Figure 10a shows its description.

In general, a function has some functional decomposition patterns to achieve it. Figure 10b shows those of a function "obtain material". Note that Figure 10b shows *is-achieved-by* relations among the functional concepts, while Figure 4 shows *is-a* relations as the definitions of them, which are independent of "how to realize them".

A super-function is decomposed into sub-functions by specifying something related to the ways to achieve it. (In the task context of the functional hierarchy understanding, the reverse operation of the functional decomposition, the information is lost.) According to what is specified, we can categorize the functional decomposition patterns as follows (the notation of the examples in the list is that super-function → sub-function + sub-function$_2$, if any):

Categories of the functional decomposition patterns (part)

- **Physical phenomena** are specified. (e.g.) "convert thermal energy of gas to kinetic energy" → "expansion of gas"
- **Target objects** of interest are specified. (e.g.) "generate water" → "generate water from steam".
- **Intermediate objects** are specified. (e.g.) "generate electricity" → "obtain kinetic energy" + "rotate"
- **Tools** are specified. The tool is an object or energy required to achieve the function besides the target objects. (e.g.) "heat" → "generate heat energy" + "give heat energy"
- **Devices** are specified. (e.g.) "separate" → "filter". Indeed the functional concept "filter" itself is a verb but it also implies use of some filters.
- **Ways** are specified. This is the case where the information is not categorized into the other categories. (e.g.) "obtain object" → "retrieve object"

Basic Procedure

The functional decomposition patterns enables the understanding system to generate super-functions from given sub-functions in a bottom-up manner. First, given a set of functions, grouping of given functions is done according to a specific condition discussed below. Next, the understanding system searches for such functional decomposition patterns that match the functions in each group.

Because many functional hierarchies can be generated from a set of functions, the knowledge for selecting super-function is need. We prepare heuristics discussed below.

Heuristics for Hierarchy Understanding

We have identified twelve heuristics shown in Table 1. These heuristics play two roles, (1)to determine preferences of super-function and (2)to specify the condition for grouping the functions. Some of them can be adjusted by users, which enables the system to generate various functional hierarchies. According to the role and the user's ability to adjust, these heuristics are categorized into four categories A,B,X and Y.

The heuristics in the categories A and B which represent the assumptions for human recognition of functional hierarchy are always assumed by the understanding system. Those in the category B play a role to determine preferences of groups of functions, that is, which groups of functions should be firstly interpreted into a super-function. For example, *the B1:serial heuristic* reflects humans understanding way based on the temporal order. *The B3:functional relations heuristic* represents a preference of super-functions supported by many functional relations.

On the other hand, the heuristics in the category X and Y are adjusted by users, which enables the system to generate various functional hierarchies.

Those in the category X determine preferences of groups of functions. The users can specify the order of applying the heuristics. For example, when a user specifies that *the X1:parallel relations* are preferred than *the X2:causal relations* (X1>X2), the understanding system firstly generates a super-function from functions in parallel-type relations, and then generates super-functions from those in causal-type relations. It means that additional functions will be integrated into the hierarchy.

The heuristics in the category Y specify the condition for grouping the functions. The users can specify the order of relaxing them (or not apply the heuristics). For example, when a user specifies that *the Y1:groups made by structure heuristic* should be relaxed after than *the Y2:groups made by energy heuristic* (Y1>Y2), the hierarchy reflecting the structure will be generated.

A setting (specification) of the orders of the heuristics provides specific criteria for generating the functional hierarchies. It can be viewed as a *viewpoint* for recognition of the functional hierarchy, and the generated hierarchy reflects the viewpoint. We are currently trying to conceptualize the viewpoints and to identify the relationship between the settings of each heuristic and the viewpoints.

Procedure based on the Heuristics

The users specify the order of applying the heuristics in the category X and the order of relaxing the heuristics in the category Y. Given the orders, grouping of functions is done according to the all criteria of the category Y specified by users. Then, in each functional group, the understanding system searches for such functional decomposition patterns that match the functions which have the preferred functional relations according to the specified order of the heuristics in the category X.

When there is no functional group to be interpreted, one of the heuristics in the category Y is relaxed according to the specified order. Then new functional groups are made, and then the super-functions are generated from them.

Table 1: Heuristics for the Hierarchy Understanding

Category A: Functional concepts heuristics (mandatory)

- *A1:Super-function heuristic.* Given a viewpoint for recognition, there is a super-function for a functional group.
- A2:*Causal relation conservation heuristic.* The causal relations among parameters are conserved in generating functional hierarchies.

Category B: Preference Heuristics (mandatory)

- ***B1**:Serial heuristic.* In the functional groups which have serial-type functional relations, the system can generate super-functions from head of chains of functions.
- ***B2**:Simultaneous heuristic.* In the functional groups which have parallel-type functional relations, the system can generate firstly a super-function from functions which have simultaneous relations than from those which have others.
- ***B3**:Functional relations heuristic.* The super-function from sub-functions which have many functional relations is preferred.

Category X: Preference Heuristics (orders are adjustable)

- ***X1**:Parallel relations preferred heuristic.* Such functional groups that have parallel-type relations are preferred.
- ***X2**:Causal relations preferred heuristic.* Such functional groups that have causal-type relations are preferred.
- ***X3**:Coverage preferred heuristic.* Such functional groups that have many functions are preferred.

Category Y: Grouping Heuristics (optional)

- ***Y1**:Groups made by **structure** heuristic.* The component of the functions in a functional group is the same from each others.
- ***Y2**:Groups made by **energy** heuristic.* The energy which the function focuses on is the same.
- ***Y3**:Groups made by **medium** heuristic.* The medium which the function focuses on is the same.
- ***Y4**:Groups made by **attribute** heuristic.* The type of functional parameters is the same.

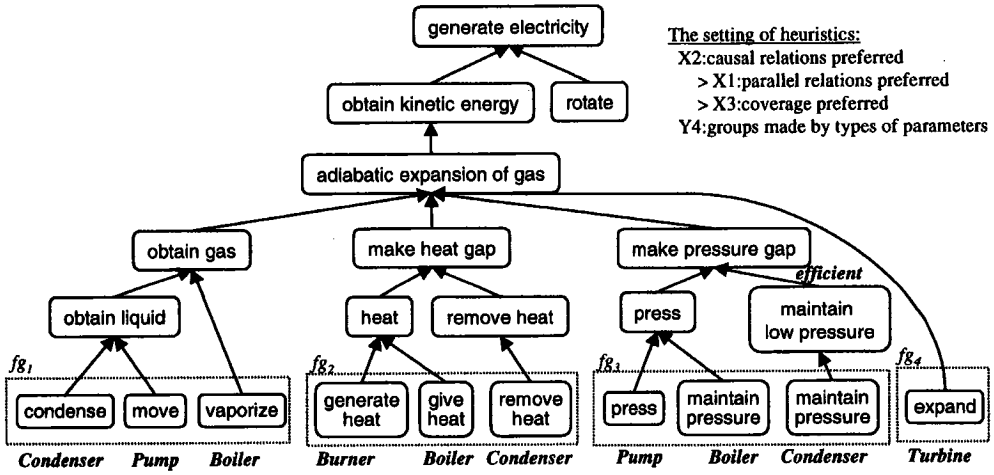


Figure 11: Example 1 of generated functional hierarchy

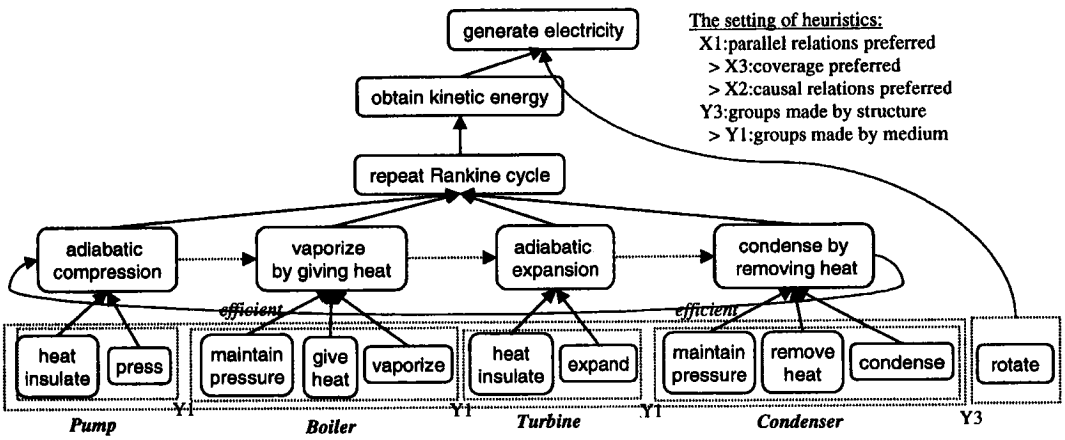


Figure 12: Example 2 of generated functional hierarchy

Example

This subsection shows three examples according to the three different settings of the heuristics.

Example (1) In this example, the settings of heuristics are "*X2:causal relations preferred > X1:parallel relations preferred > X3:coverage preferred*", and "*Y4:groups made by types of parameters*". Thus, the understanding system makes the functional groups such as the functions changing pressure-type parameters as shown in Figure 11. In the functional group fg_2, a functional decomposition pattern of "heat" (see Figure 10) matches "generate heat" and "give heat" with a proportional relation, then a super-function "heat" is generated. Next, "make heat gap" is generated from "heat" and "remove heat". Because there is a proportional-type functional relation between "generate heat" and "give heat", the system firstly generates not "make heat gap" but "heat" according to the heuristic X2, that is, the functional groups which have the causal relations are preferred.

Example (2) In this example, the settings are "*X1:parallel relations preferred > X3:coverage preferred > X2:causal relations preferred*", and "*Y3:groups made by structure > Y1:groups made by medium*". The understanding system can identify functional groups shown in Figure 12. They correspond to components (the structure among components) according to *the Y1:groups made by structure heuristic. The Y3:groups made by medium heuristic* contributes the isolation of "rotate". Firstly, according to *the X1:parallel relations preferred heuristic*, super-functions such as "adiabatic compression" and "adiabatic expansion" are generated. Next, because no functional group remains, the system relaxes the weaker heuristic *Y1*. Then, the function "repeat Rankine cycle" is generated according to *the X3: coverage heuristic*. In this result, many functions such as "adiabatic expansion" correspond

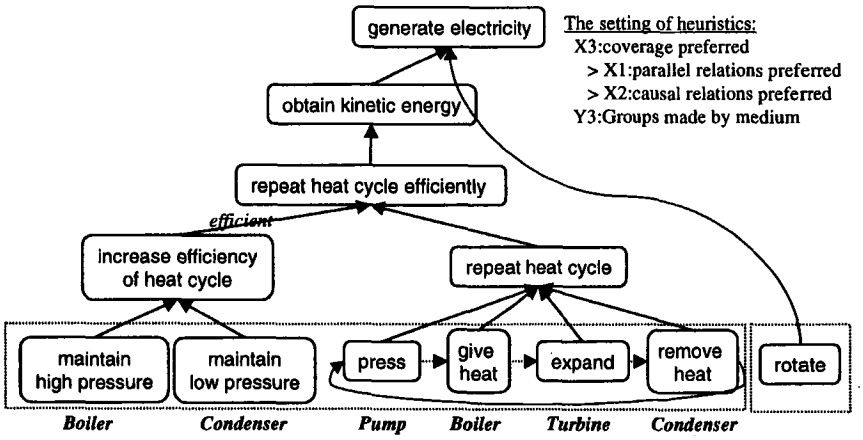


Figure 13: Example (3) of generated functional hierarchy

to structures such as the turbine according to *the Y1:groups made by structure heuristic* and *the X1:parallel relations preferred heuristic.*

Example (3) The setting of heuristics are "*X3:coverage preferred > X1:parallel relations preferred > X2:causal relations preferred*", and only "*Y3:groups made by medium*". Because *the Y1:groups made by structure heuristic* is omitted and *the X3:coverage preferred heuristic* is specified, "repeat heat cycle" is firstly generated as shown in Figure 13. According to a generic rule that a super-function can be generated from the functions which have the same type of functional relations, "increase efficiency" is generated. In this case, the crucial function "repeat heat cycle" detached from an optional function "increase efficiency" is identified according to the heuristics setting.

Related Work

Because the characteristics of other functional modeling have been discussed in section 2, we here discuss functional understanding.

The functional understanding based on FR (Thadani and Chandrasekaran 1994) uses templates of CPDs representing functional hierarchies as behavioral causal relations. Thus, functional hierarchies are directly generated from the behavioral model without the functional concepts. They are limited to those associated with structure. In contrast, our two-step decomposition of the task and the functional ontology enable the understanding system to generate various functional hierarchies by reasoning with the functional concepts according to several viewpoints in addition to structure.

In the teleological analysis proposed by de Kleer (de Kleer 1984), a few functions in the electronic circuits are defined as specific causal patterns among parameters of components (called configuration. e.g., I-LOAD function of a resistor). We define a function not as causal relations but as a result of the interpretation, thus we define more rich functional concepts such as "remove" in the is-a hierarchy. Next, his process of aggregation of local functions (called "parsing") is done by some substitution rules according to the topology of the circuits. We decompose it into two phases, that is, the functional relations understanding and the functional hierarchy understanding, and describe the decomposition patterns in terms of the general functional concepts and the types of the functional relations detached from the topology of the connections among components. Then, the understanding system can generate various hierarchies which do not correspond to the structure.

Price et al. discuss the interpretation of behavior with functional labels (Price and Pugh 1996, Snooke and Price 1997). It corresponds only to the behavior-function mapping. Thus, the functional hierarchies in (Snooke and Price 1997) always correspond to given structural hierarchies of the target system.

Summary

We have shown a functional ontology based on FBRL together with its application to functional understanding. The main points are as follows:

- **Functional Ontology.** We presented a functional ontology consisting of various meaningful functional concepts shown in Figure 4. Their definitions in terms of the mapping primitives called FTs realize maximization of their independence of behavior and structure. They are detached from functional decomposition. The functional decomposition patterns in terms of functional concepts shown in Figure 10 are independent of behavior and structure.
- **Functional understanding.** We showed a framework of functional understanding, that is, identifying functional concepts and functional hierarchies of an artifact from given structural and behavioral models of it. The

functional ontology enables the functional understanding system to generate meaningful functions from the behavior as shown in Figure 8. Moreover, the functional decomposition patterns enables the system to generate various functional hierarchies.

The functional ontology currently includes the functional concepts in fluid-related systems as discussed in Section 4. The evaluation and extension of the ontology remain as future work. We are currently investigating more details of the types of functional decomposition patterns shown in this article and implementing the understanding system. As discussed in Section 9, we are also investigating the relationship between the viewpoints for recognition of the functional hierarchies and the settings of the heuristics.

Acknowledgments

The authors would like to thank Mitsuru Ikeda for his valuable comments. This research is supported in part by the Japan Society for the Promotion of Science (JSPS-RFTF97P00701).

References

Abu-Hanna, A., and Jansweijer, W. 1994. Modeling Domain Knowledge Using Explicit Conceptualization, *IEEE Expert*, 9(5):53-63.

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.*

de Kleer, J., and 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.

Goel, A., and Chandrasekaran, B. 1989. Functional Representation of Designs and Redesign Problem Solving. In *Proc. of IJCAI-89*, 1388-1394.

Gruber, T. R. 1993. A Translation Approach to Portable Ontology Specifications, *Knowledge Acquisition*, 5(2):199-220.

Iwasaki, Y., and Chandrasekaran, B. 1992. Design Verification Through Function- and Behavior-oriented representations - bridging the gap between function and behavior -. In *Proc. of AI in Design '92*, 597-616.

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

Kitamura, Y. et al. 1996. A method of qualitative reasoning for model-based problem solving and its application to a nuclear plant. *Expert Systems with Application,* 10(3/4):441-448.

Kitamura, Y.; Ikeda, M.; and Mizoguchi, R. 1997a. A Causal Time Ontology for Qualitative Reasoning. In *Proc. of IJCAI-97*, 501-506.

Kitamura, Y., Ikeda, M., and Mizoguchi, R. 1997b. An Ontological Consideration of Causal Time in Qualitative Reasoning Systems. In *Working papers of QR-97*, 277-285.

Lind, M. 1994. Modeling Goals and Functions of Complex Industrial Plants. *Applied Artificial Intelligence*, 8:259-283.

Lee, J. 1997. Design rationale systems: understanding the issues. *IEEE Expert*, 12(3):78-85.

Mars, N. J. I. eds. 1995. *Towards Very Large Knowledge Bases*. IOS Press.

Miles, L. D. 1961. *Techniques of Value Analysis and Engineering*. McGraw-hill.

Mizoguchi, R., and Ikeda, M. 1997. Towards Ontology Engineering. In *Proc. of PACES/SPICIS '97*, 259-266.

Price, C. and Pugh, D. 1996. Interpreting Simulation with Functional Labels. In *Working papers of QR-96*, 198-204.

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

Sembugamoorthy, V., and Chandrasekaran, B. 1986. Functional Representation of Devices and Compilation of Diagnostic Problem-Solving Systems. In *Experience, memory and Reasoning*, 47-73.

Snooke N. and Price C. 1997. Hierarchical Functional Reasoning. In *Proc. of IJCAI-97 Workshop on Modeling and Reasoning about Function*, 11-22.

Umeda, Y. et al. 1990. Function, Behavior, and Structure. *AI in Engineering*, 177-193.

Tejima, N. et al. eds. 1981. Selection of functional terms and their categorization. Report 49, Society of Japanese Value Engineering (In Japanese).

Thadani, S., and Chandrasekaran, B. 1994. Constructing Functional Models of a Device from its Structural Description. In *Working papers of QR-94*, 276-285.

Vescovi, M.; Iwasaki, Y.; Fikes, R.; and Chandrasekaran, B. 1993. CFRL: A Language for Specifying the Causal Functionality of Engineered Devices. In *Proc. of AAAI-93*, 626-633.