Proc. of 2nd International Workshop on Strategic Knowledge and Concept Formation (SKCF 99), pp.181-192, 1999.

Towards Redesign based on Ontologies of Functional Concepts and Redesign Strategies

Yoshinobu Kitamura and Riichiro Mizoguchi

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

Abstract

This research aims at developing a redesign problem solver which proposes suitable modifications of a given artifact for a new requirement or an inconvenience. We concentrate on the two issues, capturing design rationales of the given artifact and organizing general redesign strategies. For the former issue, we propose an ontology of functional concepts, which enables the redesign system to identify functional structures representing a part of design rationales automatically. It particularly includes a new category of functional concepts called meta-functions for representation of dependency among functions of components. For the latter issue, we describe an ontology of the redesign strategies. Some general strategies for proposing redesign solutions in terms of the concepts in the ontology are shown. It reveals conceptualization behind the redesign strategies and then helps us specialize the Soar architecture into a universal redesign engine.

Keywords: redesign, functional representation, design rationale, modeling of strategies

1. Introduction

The redesign problem is one of the major types of design problems. Given a new requirement or an inconvenience together with structural and behavioral models of an existing artifact, the redesign problem solver proposes suitable modifications free from the inconvenience. In order to realize such a redesign solution proposing system, we concentrate on the following two issues, that is, capturing design rationales of the given artifacts and organizing general redesign strategies.

The former issue is concerned with functional understanding of artifacts, because functionality of artifacts represents a part of the design rationale [Chandrasekaran *et al.*, 1993; Lee 1997]. It is essential for redesign of an existing artifact to understand its functional structure in order to consider the intention of the original design [Goel and Chandrasekaran, 1989]. Our goal here is to identify functional structures of the given artifacts automatically, called *functional understanding task*. We focus two problems here. One is how to limit the search space at the functional level, because human uses a large number of functional concepts without their operational definitions as discussed in Value Engineering research [Miles, 1961; Tejima *et al.*, 1981]. The other problem in functional understanding for redesign is that the study of function is still premature. Although many researchers have investigated topic of what a function is, they treat a function as a kind of abstraction of behavior of the components [Umeda *et al.*,1990; Lind,1994; Sasajima *et al.*,1995]. Such functions represent no information of dependency among functions.

The latter issue in redesign is how to organize general redesign strategies. We view a process of the redesign task as a search in the design problem-solving space like that in Soar architecture [Newell, 1990]. Our goal here is to specialize the Soar architecture in redesign and to specify the problem-solving space and strategies at the redesign task level. The conceptualization of knowledge as well as strategies is usually left implicit. This has

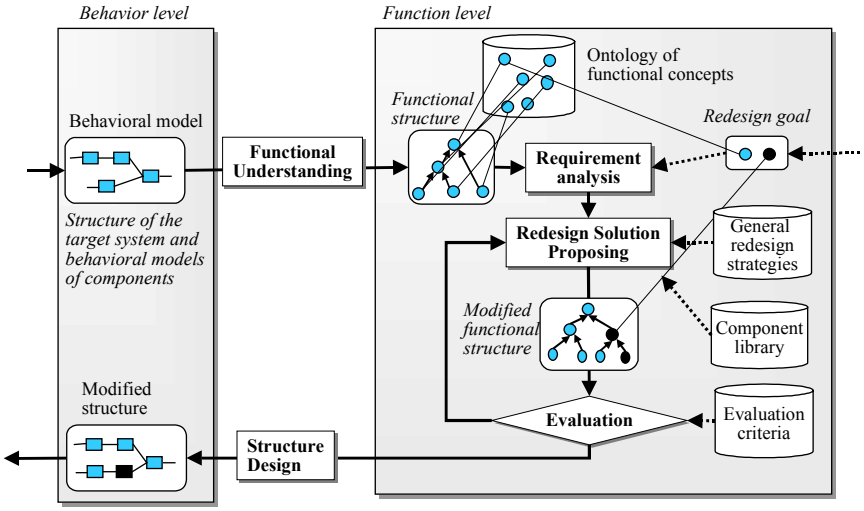

Figure 1: The framework of our redesign support system

been a major cause of the non-reusability of knowledge. It requires such a conceptual vocabulary for redesign strategies independent of the target domain.

We are tackling these two issues on the basis of Ontological Engineering [Mizoguchi and Ikeda, 1997], aiming at explicit specification of conceptualization of functional concepts and redesign strategies. For the first issue, functional understanding, we identify an ontology of functional concepts of artifacts, which provides a rich vocabulary for functional representation. The ontology plays a role to limit the search space in functional understanding and to screen out meaningless functional interpretations.

The ontology includes a new category of functional concepts named meta-function in order to represent dependency among functions. A meta-function represents a role played by a function for another function in order to make the whole system work collaboratively.

For the second issue, we are describing an ontology of a redesign task, which provides a vocabulary for representation of redesign strategies. We describe some redesign strategies for the redesign solution proposing subtasks in terms of the ontology of redesign task.

In this paper, we firstly overview the framework of our redesign support system including functional understanding. Next, the ontology of functional concepts for functional understanding and redesign is discussed. Section 4 describes the process of functional understanding. Next, description of the redesign strategies is discussed. Section 6 overviews an ontology of redesign strategies. The contribution of this work by comparison with the related work is also discussed.

2. Framework of a Redesign Support System

As Figure 1 shows, our redesign support system consists of a functional understanding module, a requirement analysis module, a redesign solution proposing module, and evaluation module. Given behavioral and structural models of the target system, the functional understanding module firstly generates its functional interpretations according to the ontology of functional concepts. It enables us to uncover the intention of the designer behind the given initial design. Next, the requirement analysis generates the initial states for the redesign module from the given redesign goal and the generated functional structure. This task includes diagnostic reasoning in order to identify the function to be improved. Then, the redesign solution proposing module proposes how to change the target system (redesign solutions). Its process can be viewed as search of operators which can change the current states to the goal states. The initial goal is decomposed into sub-goals according to redesign strategies called ways of redesign goal achievement. Lastly, the redesign solutions are evaluated according to the evaluation criteria.

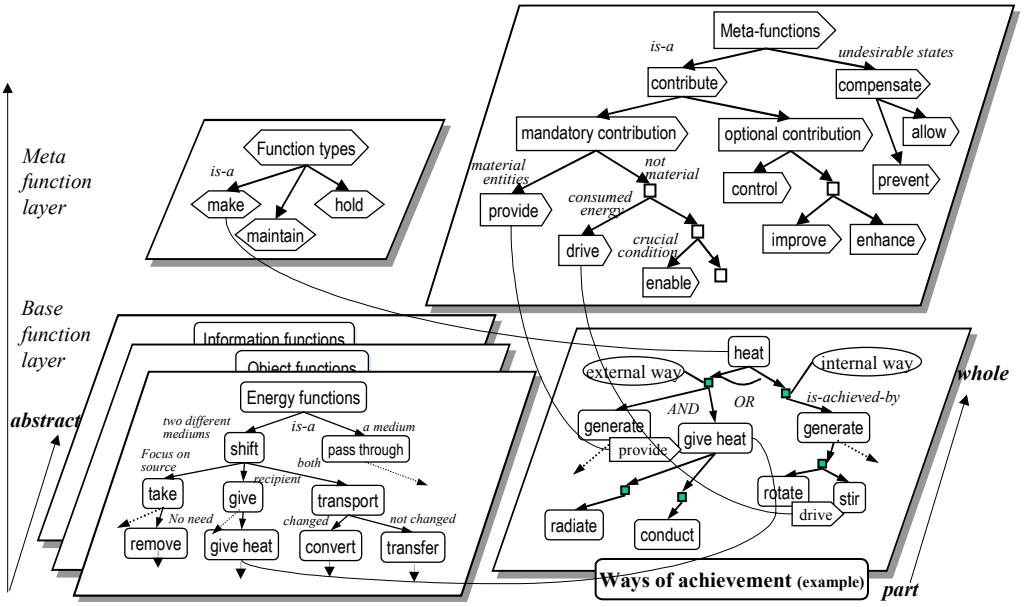

Figure 2: The four spaces of the ontology of functional concepts (part)

3. Ontology 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.

3.1. Base-functions

A base-function of a component is defined as a result of interpretation of a behavior of the component under an intended goal [Sasajima *et al.*, 1995]. A base-function of a component can be represented by a transitive verb of which grammatical subject is the component and of which grammatical objects are the incoming or outgoing entities of the component. Although a function depends on the context, the description itself should be local. A behavior 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 base-functions organized in an is-a hierarchy with clues of classification. A base function is defined by conditions of 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) [Sasajima *et al.*, 1995]. 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 components), 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 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 as the ***amount function*** which changes

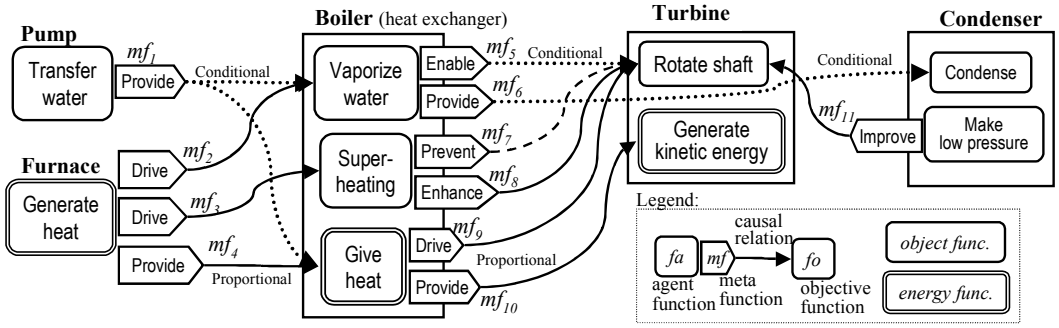

Figure 3: Meta-functions in a power plant (part)

the amount of the target objects or changes the kind of the objects. The other is called as the ***attribute function*** which 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 component. For example, P-Focus specifies not concrete location but abstract interaction with the neighboring components. The realization is represented by the ways of achievement shown in Figure 2d.

3.2. Function Types

The function types represent the types of goal achieved by the function [Keuneke, 1991]. We have redefined the function type as "ToMake", "ToMaintain", and "ToHold". For example, consider two components, an air-conditioner (as a heating device) and a heater, having the same function "to give heat". The former keeps the goal temperature of a room. The latter does not. These are said to be "ToMaintain" and "ToMake", respectively.

3.3. 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-function refers not changes of objects of these components but functions of the components, 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 Figure 2c (an is-a hierarchy). Figure 3 shows examples of the meta-functions $\{mf_1 \ldots mf_{11}\}$ in a simple model of a power plant. Note that the furnace which is a sub-component of a 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 [Kitamura and Mizoguchi, 1999a].

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 a narrower meaning of ToEnable by excepting 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) which will be a part of the output objects 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 that *intentionally* consumed by another function f_t (called *driving energy*), 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 which 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., faults) is caused in a function f_{t2} when a function f_a is not achieved, the function f_a is said to have a meta-function "to prevent malfunction of f_{t2}". 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" of the condenser contributes to the efficiency of the "to rotate" function (see mf_{11}) without increment of input energy. On the other hand, the "to super-heat" function of the boiler optionally increases the amount of the input energy (mf_8).

ToControl: When a function f_a regularizes the behavior of f_t, its meta-function is said to be "to control f_t". For example, consider a control valve which 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).

3.4. Ways of Functional Achievement

A base-function f_u can be achieved by the different groups of sub-functions. We call a group of sub-functions $\{f_1, \ldots f_n\}$ 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.

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".

4. Functional Understanding

The functional understanding task is to identify functional structures of an artifact from the given structural information and behavioral models of components. Although such task in principle difficult because the search space of function is huge, the ontology plays a role to

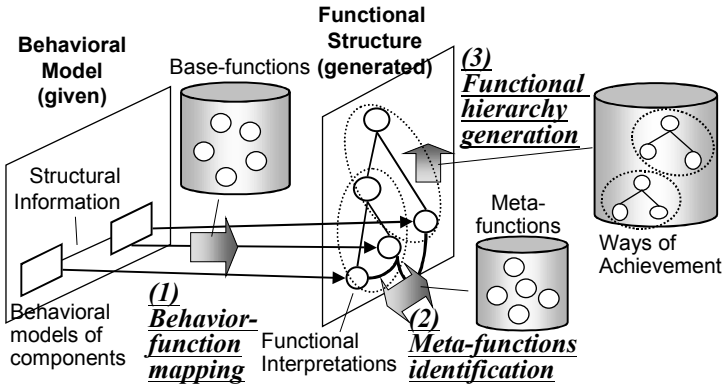

Figure 4: Three steps of functional understanding

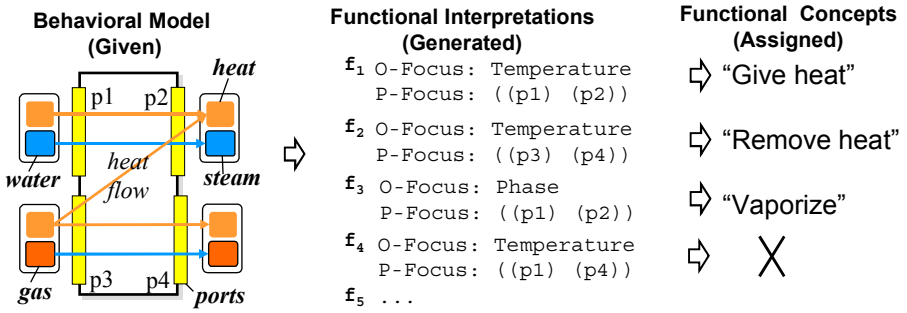

Figure 5: Behavior-function mapping of a boiler

limit the search space. The ontology provides such primitives in the functional space that are targets in the mapping, and screens out meaningless functional interpretations.

The process of identification shown in Figure 4 consists of the following three steps; behavior-function mapping, identification of meta-functions among base-functions, and generation of functional hierarchies. See [Kitamura and Mizoguchi, 1998] for the detail of the identification of functional structures.

4.1. Behavior-function mapping

Firstly, the understanding system exhaustively generates candidates of base-functions to be performed by each component as all tuples of possible values of FTs context-independently. For example, in the case of the boiler shown in Figure 5, the system generates a functional interpretation f_3 which consists of O-Focus on the "phase" parameter and P-Focus on the inlet water and the outlet steam.

Then, the understanding system screens out meaningless ones by matching them with the base-functions in the ontology. Such functional interpretations that match with no concept in the ontology are screened out as a meaningless interpretation assuming the completeness of the ontology in the functional space. In Figure 5, the functional interpretation f_3 is successfully matched with a functional concept "vaporize". In contrast, f_4 is screened out as a meaningless interpretation. Although many candidates of the functional interpretations remain, plausible functional interpretations are identified by the following steps.

4.2. Identifying meta-functions

Secondly, the understanding system identifies the function types and meta-functions between the given a pair of base-functions according to the definitions of meta-functions shown in Figure 2c. The identification algorithm is described in [Kitamura and Mizoguchi, 1999a]. It includes our qualitative reasoning engine [Kitamura *et al.*, 1997] and a diagnostic engine [Kitamura and Mizoguchi, 1999b].

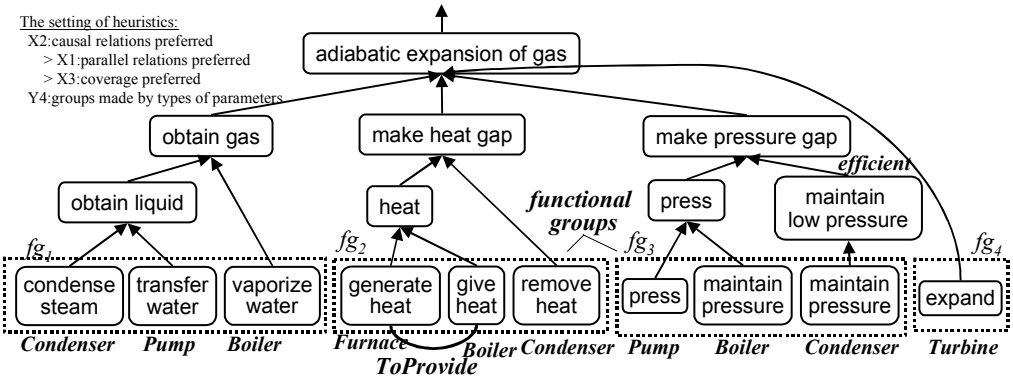

Figure 6: An example of generated functional hierarchy

For example, imagine that we are given "to generate heat" function of the furnace and "to vaporize water" function of the boiler in Figure 3. Firstly, causal relations between the functions are checked. Because there is a mandatory causal relation from the focused parameter of the heat generation (the amount of the heat energy of the combustion gas) to the focused parameter of the vaporization (the amount of the steam), then the heat generation is the agent function and the vaporization is the target function.

Next, the conditions of ToProvide are checked. Because the focused entity of the furnace (the heat energy of the combustion gas) is not material of the focused entity of the target function (the inlet water), the meta-function is not ToProvide.

Next, the conditions for ToDrive are checked. Firstly, the focused entity of the agent function (the heat energy) is energy and the focused entity of the target function (the steam) is an object. Secondly, the causal relation between the amount of the heat energy and the amount of the steam is basically proportional. Thirdly, the amount of the heat energy is reduced by the boiler. Lastly, the focused attribute of the agent function is the amount. Then, the meta-function between them is ToDrive (mf_2 in Figure 3).

4.3. Generation of Functional Hierarchies

The hierarchies of the functions are identified in a bottom-up manner according to the knowledge of the way of functional achievement shown in Figure 2d. First, given a set of base-functions, grouping of given functions is done according to the grouping heuristics. Next, the understanding system searches for such way of achievement that match the functions in each group. Although many functional hierarchies can be generated from a set of functions, preference heuristics enables the system to select super-functions. We have identified twelve heuristics for the two purposes. Because each type of meta-functions has own strength to make the functional groups, the grouping can be done according to the types of the meta-functions among them. Some of heuristics can be adjusted by users, which enables the system to generate various functional hierarchies. The details of the heuristics are described in [Kitamura and Mizoguchi, 1998].

Figure 6 shows an example of the functional hierarchy generated by the system. Firstly, the system makes the functional groups such as the functions changing pressure-type parameters (fg_3 in Figure 6) according to the specified grouping heuristics "*groups made by types of parameters*". Next, in the functional group fg_2, the external way of "to heat" in Figure 2d matches "to generate heat" and "to give heat" with a ToProvide meta-function, then a super-function "to heat" is generated. Next, "make heat gap" is generated from "to heat" and "to remove heat". Because there is a meta-function between "to generate heat" and "to give heat", the system firstly generates not "make heat gap" but "heat" according to the specified heuristic X_2, that is, the functional groups which have the causal relations and meta-functions are preferred.

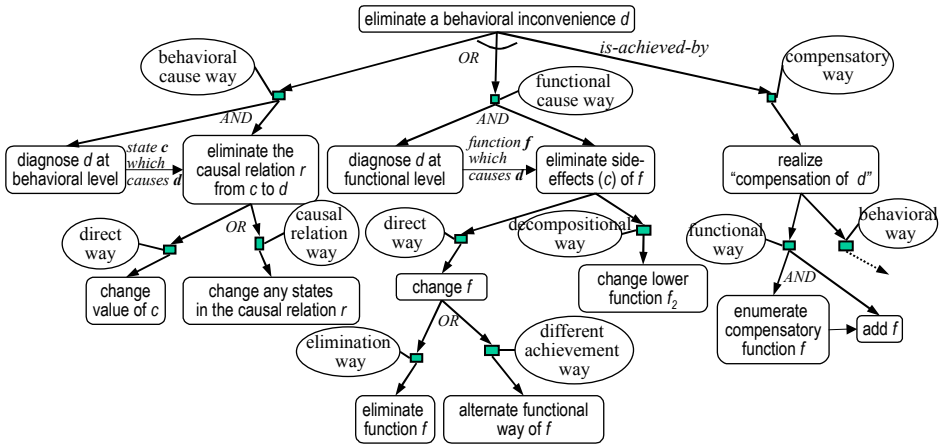

Figure 7. Examples of ways to achieve a redesign goal

When the user changes the order of applying the heuristics or relaxing, the different functional hierarchies are generated. The user's specification of heuristics can be viewed as a viewpoint for recognition of the target system, and the generated hierarchy reflects the viewpoint. Some different hierarchies are shown in [Kitamura and Mizoguchi 1998].

5. Ways of Redesign Goal Achievement

In this section, we overview the knowledge for redesign, especially for proposing redesign solutions. The knowledge can be categorized into task knowledge and domain knowledge of the target products. The former includes ways of redesign goal achievement discussed in the following paragraphs. The latter includes the product models (which consists of structural, behavioral and functional models) and the ways of functional achievement discussed in Section 3. Such knowledge about achievement of functions enables the redesign system to find alternative ways to achieve a function. For example, when a function in the current product model causes unexpected phenomena, the redesign system can explore such an alternative way to achieve the function without the phenomena.

We describe general knowledge about "ways" to achieve the redesign goal by some sub-goals, called *ways of redesign goal achievement*. Each of them consists of a (super-)goal and sub-goals which can achieve the super-goal. It can be viewed as a kind of redesign strategies, which can decompose current goal into specific sub-goals in order to propose redesign solutions free from the given inconvenience. The inconvenience is determined according to specific criteria when the product models and evaluation criteria are given.

For example, Figure 7 shows some ways to eliminate a behavioral inconvenience. In the case of the functional way of the elimination, the top-goal is decomposed into the two sub-goals, that is, "to identify function *f* which causes the inconvenience" (denoted as "diagnose the inconvenience at functional level") and "to eliminate the side-effect of *f*". The former can be achieved by the diagnostic reasoning (not shown in the figure). The latter can be achieved by either of the alternative sub-goals, that is, "to change the function *f*" or "to change a sub-function *f2* of *f*". The former can be achieved by either "to eliminate function *f* itself" or "to alternate functional way of *f*". On the other hand, in the case of the compensatory way, the same top-goal is decomposed into "to enumerate function *f* which compensates *d* (called compensatory function)" and "add function *f*".

Such ways to achieve the redesign goal are independent of the specific target systems and attributes. Some of redesign strategies, however, are dependent on the specific attributes. For example, the goal "to fire two functions f_1 and f_2 simultaneously" can be decomposed into the two sub-goals; "to add a function f_3 'to check if f_1 is working'" and "to add a meta-function 'to allow f_3 to fire f_2'" where f_1 is an interval function and f_2 is an instant function.

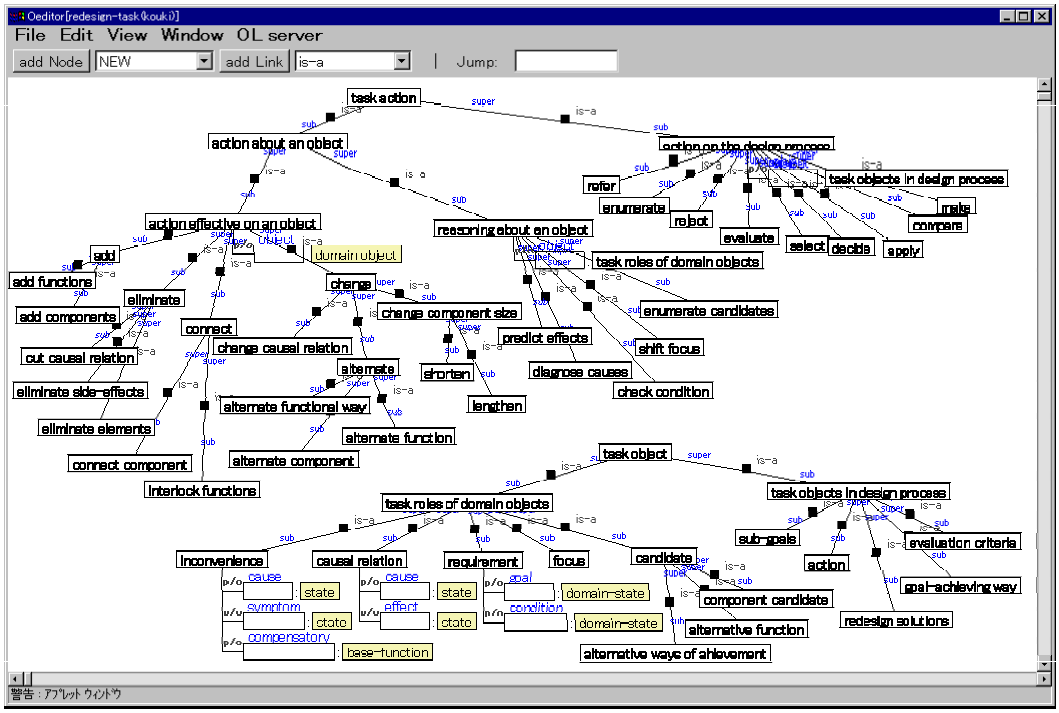

Figure 8. An ontology for redesign strategies (not exhaustive)

This redesign strategy is specific to the attribute “time”. Nevertheless, it is still general in the sense that it does not depend on the meaning of the functions.

6. Ontology for Redesign Strategies

We are currently investigating an ontology for redesign strategies, especially for the ways of redesign goal achievement. It provides vocabulary for representation of the ways of redesign goal achievement and specifies the redesign problem solving space, and hence we can explicitly specialize Soar to redesign tasks. It also provides rationales for knowledge engineer who describes such knowledge, and then it facilitates their description and makes the knowledge reusable.

This ontology is a kind of a task ontology about the redesign solution proposing task. In general, a task ontology represents the process of the problem solving, which consists of ***task action*** and ***task object***. A task action represents a unit (a primitive) in the problem solving process, which usually corresponds to a verb in sentences representing the process of the problem solving in a natural language. A task object represents an object of a task action, which corresponds to a noun.

Figure 8 shows is-a hierarchies of a part of the ontology. The task actions in our ontology are categorized into **action about an object** and **action on the design process**. The former is further divided into **action effective on an object** and **reasoning about an object**, while the latter represents a kind of meta-action which affects states in the design processes such as **to enumerate possible next actions** and **to select a next action**. Action is the example of design task objects.

Action effective on an object causes a change in the target products such as *to shorten length* and *to add a component*. Their objects are domain objects in the domain ontology. **Reasoning about an object** has subclasses such as *to diagnose causes of inconvenience* and *to predict effect*. Their objects are roles of domain objects in the redesign task, called task roles, such as *inconvenience* and *effect*.

The ways of redesign goal achievement shown in Figure 7 can be described in terms of concepts in the ontology for redesign strategies. The conceptualization of knowledge as well as strategies is usually left implicit. This has been a major cause of the non-reusability of knowledge. The redesign strategy ontology reveals such conceptualization as shown in Figure 8 and provides people with the basic information as a first step towards a shared understanding of the task. Furthermore, it helps us specialize Soar architecture into a universal redesign engine. We first doublecheck the appropriateness and sufficiency of the concepts by investigating the ontology, and then we specialize some of the Soar terms as well as search control knowledge and selection space in terms of the ontology.

7. Related Work

Ontology of functional concepts:

The functions in [de Kleer, 1984; Umeda *et al.*, 1990; Lind, 1994; Sasajima *et al.*, 1995; Chandrasekaran and Josephson, 1996] represent abstracted 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 [Sembugamoorthy and Chandrasekaran, 1986; Vescovi *et al.,* 1993], 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 [Vescovi *et al.*, 1993] represents causal relations among functions which correspond to relation among functions on the function layer in our framework. Lind categorizes such relations into Connection, Condition and Achieve [Lind, 1994]. Rieger identifies "enablement" as a type of the causal relation between states and action [Rieger and Grinberg, 1977]. 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 [Hodges, 1992], the sets of "primitives of behavior" are proposed. Lind identifies a few general functions such as "storage of energy" which are categorized into multiple levels [Lind, 1994]. We added more intention-rich concepts such as "remove" with unnecessary intention to the set of the base functions and organized in is-a and part-of hierarchy. 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.

As types of base-functions, we redefine ToMake and ToMaintain [Keuneke, 1991]. Our other functional type "ToHold" is similar to "ToAllow" identified in [Bonnet, 1992].

The consolidation theory [Bylander and Chandrasekaran, 1985] tries to capture the general patterns of aggregation of the system. In a literature on design, many general "patterns" of synthesis are proposed (e.g., [Bradshaw and Young, 1991; Bhatta and Goel, 1997]). Our general ways of functional 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 [Takeda *et al*., 1990]. They enable the redesign system to facilitate the smooth interaction between models at the structural and functional levels.

Ontology for redesign strategies:

Analyses of (re)design activity have identified the top-level task structures of (re)design [Brazier, et al. 1994; Chandrasekaran, 1990; Goel and Chandrasekaran, 1989; Gero, 1990; Pos *et al.*, 1997]. Especially, the research efforts from the viewpoint of the task-oriented methodologies for knowledge-based systems identify reusable problem-solving methods in (re)design [Chandrasekaran, 1990; Pos *et al.*, 1997]. Although our ontology for redesign

strategies is a kind of a task ontology of redesign solution proposing task which is a subtask of redesign, we aim at conceptualization of the redesign strategies.

DSPL shown in [Brown and Chandrasekaran, 1989] is a language for knowledge of routine design including *design plans* which represent precompiled sequences of design actions. Our aim here is not knowledge representation scheme but general redesign strategies (plans) and articulation of concepts in such knowledge.

In [Goel and Chandrasekaran, 1989], the redesign solutions are categorized into *corrective* one, or *compensatory* one, and the former task is decomposed into *diagnosis* and *repair* subtasks. We identified richer redesign strategies including them. In our redesign strategies, the corrective redesign is described as "behavioral way" to eliminate a given inconvenience.

Designer-Soar mentioned in [Newell, 1990] is a Soar-based system to design algorithms. It uses a general set of functional operations such as generate, select and test. Johnson *et al.* shows an architecture which introduces task-specific structures into the Soar architecture [Johnson *et al.,* 1990]. Our ontology aims at richer operators and strategies which are specific to redesign of engineering products.

8. Summary

We proposed an ontology of functional concepts including eight types of the meta-functions, which represent the types of collaboration and dependency among functions of components. They contribute to functional understanding task which identifies functional structures automatically from given structural and behavioral models. We also discussed redesign strategies as ways of redesign goals achievement (and decomposition) and an ontology for representation of them.

Currently, our ontology of functional concepts assumes the existence of something flowing (or transferred) among components which carries energy (called objects). Then, it covers functions in fluid-related plants including power plants, chemical plants, and manufacturing processes and does not cover mechanical phenomena. An investigation on the difference of functional concepts in different domains is in progress.

Acknowledgments. The authors would like to thank Kouki Higashide, Toshio Ueda, and Koji Namba for their contributions to this work. The authors are grateful to Mitsuru Ikeda for his valuable comments. This research is supported in part by the Japan Society for the Promotion of Science (JSPS-RFTF97P00701).

References

Bhatta, S. R., and Goel, A. K. 1997. A functional theory of design patterns. In Proc. of IJCAI-97, 294-300.

Bonnet, J. C. 1992. Towards a formal representation of device functionality. TR 92-54, Knowledge Systems Laboratory, Stanford University.

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

Brazier, *et al.* 1994. On formal specification of design tasks. *AI in Design '9*4, 535-552.

Brown, D. C. and Chadrasekaran, B. 1989. *Design problem solving: knowledge structures and control strategies*, Pitman Pub. and Morgan Kaufmann.

Bylander, T., and Chandrasekaran, B. 1985. Understanding behavior using consolidation, *Proc. of IJCAI-85*, 450-454.

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

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. 1984. How circuits work, *Artificial Intelligence* 24:205-280.

Gero, J.S. 1990: Design prototypes: a knowledge representation schema for design, AI Magazine, 11(4), 26-36. 1990

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

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

Johnson, T.R., Smith, J.W., and Chanrasekaran, B. 1990. Task-specific architectures for flexible systems. *The Soar Papers*, 1004-1026, MIT Press.

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

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

Kitamura, Y., and Mizoguchi, R. 1998. Functional ontology for functional understanding, Papers of *Twelfth International Workshop on Qualitative Reasoning (QR-98)*, 77-87.

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

Kitamura, Y., and Mizoguchi, R. 1999b. An Ontological Analysis of Fault Process and Category of Faults, *Proc. of Int'l Workshop on Principles of Diagnosis (DX-99),* 118-128.

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.

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.

Newell, A. 1990. *Unified theories of cognition*, Harvard Univ. Press.

Pos, A., Wijngaards, N.J.E, Straatman, R., and REVISE, 1997. Choices in problem solving methods for redesign. Technical Report IR-419, Vrije Universiteit Amsterdam.

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

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.

Takeda, H.; Veerkamp, P.; Tomiyama, T.; and Yoshikawa, H. 1990. Modeling design processes. *AI Magazine*, 11(4), 37-48.

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

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

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.