An Ontological Analysis of Faults

Yoshinobu Kitamura
and
Riichiro Mizoguchi

AI-TR-99-2
Received March 25, 1999

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

Abstract

In an interactive system for problem solving, its transparency concerning capability and limitation is critical for ease of understanding to facilitate the utility of the system. Our aim here is explication of capabilities and limitations of di-agnostic systems. We focus on the process in which faults are induced and establish an ontology of fault including several categories of fault. It provides us with a vocabulary for specifying the scope of diagnosis. Firstly, the ontology enables us to specify capability of diagnostic systems for transparency to users. Next, we design an interactive system to enumerate the “deeper causes” of a malfunction. The ontology enables the human users to control the diagnostic process in a plausible-first way interactively

1 Introduction

In an interactive system for problem solving, its transparency is critical for ease of understanding to facilitate the utility of the system. Among variety of understanding, the specification of system's capability of problem solving as well as its limitations is of importance. No system is free from an assumption on which its performance largely depends. Nevertheless, such an assumption is rarely explicit, which has been one of the serious causes of low usability and extensibility of the systems.

Our goal here is the explication of system's assumption of model-based diagnostic systems [Hamscher *et al.*, 1992] through ontological engineering [Mars, 1995; Mizoguchi and Ikeda, 1997]. Although there has been a lot of research on capability of model-based diagnostic systems, most of them are investigation from the logic point of views such as hypothetical reasoning [Poole, 1989; Console and Torasso, 1991] and that about multiple faults [de Kleer and Williams, 1987; Tatar, 1996].

An exception is the pioneer work done by Davis [1984] in which he points out that capability of diagnostic systems are limited within the scope specified by their assumptions about the target object and physical phenomena of interest. For example, many of the constraint-based diagnostic systems (e.g., GDE [de Kleer and Williams, 1987]) do not deal with faults caused by topological change among the components.

Furthermore, while many of the systems identify malfunctioning components as the cause of a given symptom, the deeper causes of the malfunction are left unknown (see Section 2 for an example). The deeper causes explain how the malfunction occurred and are crucial to repair the target system completely in order to prevent repeats of the same fault as pointed out in [Tatar, 1996].

Nevertheless, little analysis of concepts for specifying such limitation of diagnosing capability such as in [Davis 1984; Struss, 1992a] has been done to date and the many of the limitations and characteristics are left implicit.

In this paper, we discuss an ontology of fault focusing the "fault process" in which faults are induced, while many of the constraint-based diagnosis cope with only the resultant states of the fault process. We define concepts related to the fault process such as *absolute cause of fault* and *external influence*. On the basis of these investigation, we identify several categories of fault such as *structural fault* and *spatial propagation fault.*

The ontology of fault provides us with a conceptual vocabulary to make the scope of diagnosis explicit. Firstly, it enables us to specify the capability of the diagnostic systems, which makes the systems transparent to users who want to know what type of fault the system can diagnose when they select one. Taking GDE as an example, we will show it can diagnose rather limited type of faults.

We next design a fault model and a reasoning method to enumerate the deeper causes of malfunction. The ontology enables the human users to control the diagnostic process in a plausible-first way by interactively relaxing the diagnostic assumptions in terms of well-defined concepts. The method complements a conventional constraint-based system collaboratively.

2. An ontology of fault

2.1 Motivation and fault process

Let us explain our motivation and main idea taking a turbine system fault as an example shown in Figure 1. The story is as follows: The fault has been initiated by the quality fading of the inner pipe of super-heater due to the elapse effect. Then some pieces became to exist in the super-heater, then flew to the turbine chamber and collided with the turbine blade to break it. Then, the shape of the blade deformed to reduce the torque and then the number of rotation of the shaft decreased to finally reduce the energy generated by the generator which is observed as a symptom.

Many of the diagnostic systems try to find a faulty component by reasoning about the parameter values in a resultant state after the fault has occurred and will identify only the turbine in this example as the cause of the symptom. We can say that they do not reason about events such as breakage occurred in the process to the malfunction and hence cannot identify the deeper causes such as corrosion in the super-heater and thus the existing faulty component such as the super-heater.

We call the former state after the fault has occurred ***after-fault state***[t47] in ***after-fault time***[t30] (italic represents a concept shown in Figure 2 with own ID number headed by 't'). The latter process, which is a chain of abnormal events, is called ***fault process***[t45] in ***fault process time***[t29].

In the following, we analyze the *fault process*[t45] and identify concepts explaining it such as *fault events*[t43] and *absolute cause of the fault*[t34] shown in Figures 1 and 2.

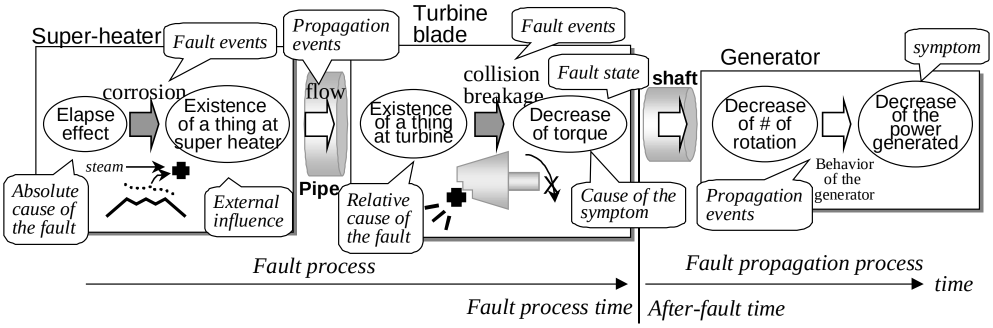


Figure 1: Example of a fault process

2.2 Concepts for the fault process

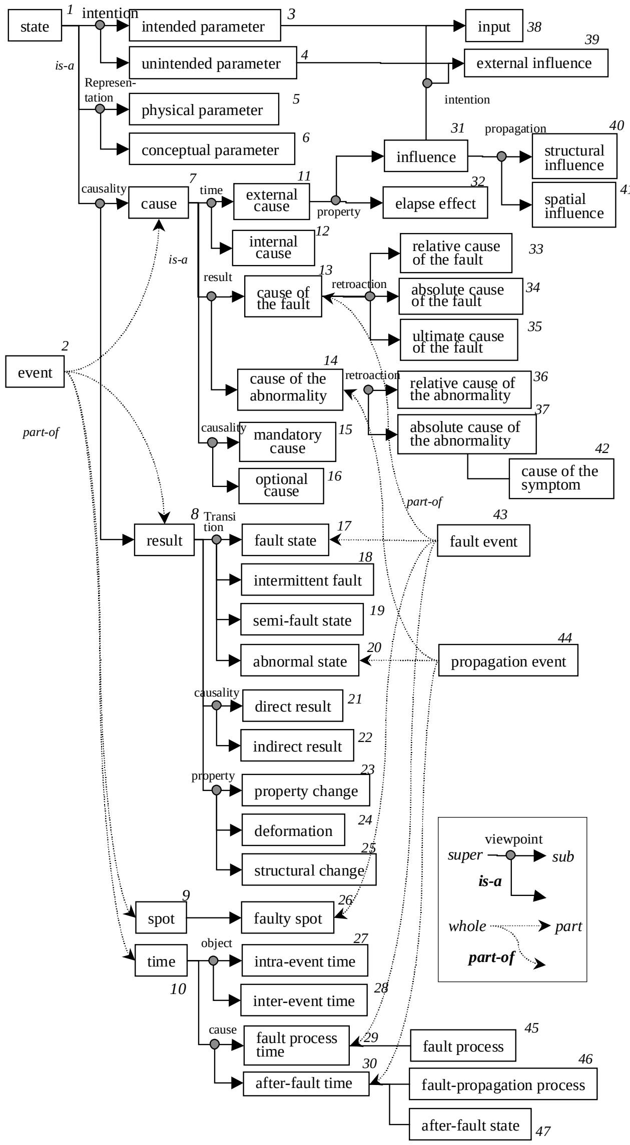
A ***state***[t1] represents values of one or more parameters. A parameter representing a state intended by the designer (i.e., parameters in the model of the correct behavior) is called an ***intended parameter***[t3] and others ***unintended parameter***[t4]. A parameter corresponding directly to physical value is called a ***physical parameter***[t5] and one which does not is called a ***conceptual parameter***[t6] such as "existence" of a thing.

We consider a *fault process*[t45] is a chain of ***events***[t2], state changes of components initiated by a certain event. An event associated with each component is represented by a quadruple, <***cause***[t7], ***result***[t8], ***the spot***[t9], ***time***[t10]>. A cause of the state change of a component includes ***influence***[t31] from outside, ***elapse effect***[t32] (e.g., the case of corrosion). Influence is further divided into ***input***[t38] and ***external influence***[t39] according to if it is intended or not and divided into ***structural influence***[t40] and ***spatial influence***[t41] according to the way of influencing.

A ***fault event***[t43] is defined as an irreversible state change of a component and the state of the *cause*[t7] (it is an *influence*[t31] in many cases) is called ***cause of the fault***[t13] and state of the *result*[t8] is called a ***fault state***[t17]. Further, we call the component in trouble ***faulty spot***[t26]. If the state change is reversible, then we call it ***semi-fault state***[t19]. If the internal state does not change, we call it ***abnormal state***[t20] caused by ***propagation event***[t44] which means the output of the component is abnormal only because of the propagation of abnormal input (e.g., "flow" event in pipe and the behavior of the turbine in Figure 1).

The whole fault process of a system under diagnosis is viewed as a chain of *fault events*[t43] and *propagation events*[t44]. That is, *fault events* induce other *fault events* or *propagation events* ending up with symptoms which are observed. The *cause of the fault*[t13] of the upper most *fault event* in the *fault process* within the model of the system is called ***absolute cause of the fault***[t34]. In general, *absolute cause of the fault* is *elapse effect*[t32] (the case of Figure 1) or *influence*[t31] coming from the outside of the current model. In the latter case, there exists at least one cause related to outside (or in the environment) of the system under consideration and it is called ***ultimate cause of the fault***[t35].

On the other hand, the cause of *propagation event*[t44] is called ***cause of the abnormality***[t14] and the upper most cause in the chain of *propagation event*s is called ***absolute cause of the abnormality***[t37] (the decrease of torque). Time passing up to the *absolute cause of the abnormality* is called ***fault process time***[t29] and that after it is called ***after-fault time***[t30]. While cause of the fault explains causes of the *fault event*s during the *fault process time*[t29], *cause of the abnormality* does states where the abnormality occurs during the *after-fault time*[t30]. It can be said that an *absolute cause of the abnormality*[t37] is a ***cause of the symptom***[t42] which explains observed symptoms.



2.3 Classes of fault

We also identify categories of fault by classifying faults with the help of concepts discussed thus far. Figure 3 shows all of them. Associated with them in italic are examples and ID numbers headed by 'f'. For example, stick of a valve, which is a malfunction of a component and is represented in terms of an *intended parameter*[t3], is classified into ***negated normal behavior fault***[f13] because it is represented by negating the constraints representing the correct behavior. On the other hand, faults which need an *unintended parameter*[t4] to represent like "contamination" and "adhesion" are classified into ***unintended parameter fault***[f14]. Faults represented by *physical parameters*[t5] are called ***parametric fault***[f15] and those require *conceptual parameters*[t6] to represent them like smudge and breakage are called ***non-parametric fault***[f16]. Concerning the resulting state, for example, ***property fault***[f10] which represents *property change*[t23] of components such as quality and strength, ***shape fault***[f11] which represents *deformation*[t24] of components such as breakage, ***structural fault***[f12] which represents change of location or that of topological structure (*structural change*[t25]) such as leakage and touch according to the property which changes.

3. Capability of diagnostic systems

Let us describe capability of GDE [de Kleer and Williams, 1987]. Concerning the result of diagnosis, it tries to identify malfunctioning components (e.g., the turbine in Figure 1) and abnormal values in the state when the symptom is observed (the decrease of the torque). Thus, we can say that “causes of fault” that GDE can identify are *faulty spots*[t26] and *cause*s *of the symptom*[t42] in *after-fault state*[t47] and **not** (*absolute) causes of the fault*[t13] representing the deeper causes of faulty spots (existence of a thing at the turbine and elapse effect of the super-heater).

Concerning the scope of the fault, GDE can be said to treat *negated normal behavior fault*[f13] because it defines a fault as “system behavior different from the normal one”. Since the model in GDE is composed of the parameters representing the correct behavior (*intended parameters*[t3]) and directly corresponding to the physical value (*physical parameters*[t5]), it cannot identify *unintended parameter fault*[f14] or *non-parametric fault*[f16] (e.g., corrosion).

Further, GDE assumes the topological structure of the target system does not change, and hence it cannot deal with the abnormal relations between components in the process of the *structural fault*[f12], *external influence fault*[f22] and *spatial propagation fault*[f24]. Such a situation is sometimes misidentified as pointed out in [Davis, 1984; Tatar, 1996]. In the example shown in Figure 1, malfunction of the super-heater cannot be detected, because the “collision” is an *external influence fault*[f22].

4. Step-wise reasoning for deeper causes

The ontology of fault discussed thus far suggests us the necessity of creating a reasoning system which can deal with wider scope of fault, that is, one covering so-called “deeper causes”, such as *external influence fault* and *spatial propagation fault*. It is not realistic to expect a fully automatic diagnostic system for such “deeper causes” because of the almost infinite search space.

Our approach is (1) a fault model as specification of the wider search space and (2) interactive system with human experts who can control the direction to pursue with the help of well-conceptualized types of fault.

4.1 A model for search space specification

In order to reason about the deeper causes in *fault process time*[t29], we have developed a model of the *fault process*[t45]. The model is composed of (1) an object model and (2) a fault event model. The former is a model for anything which can be said to be faulty (called *objects*) including components as functional and structural devices like turbine and pipes and medium such as liquid and gas.

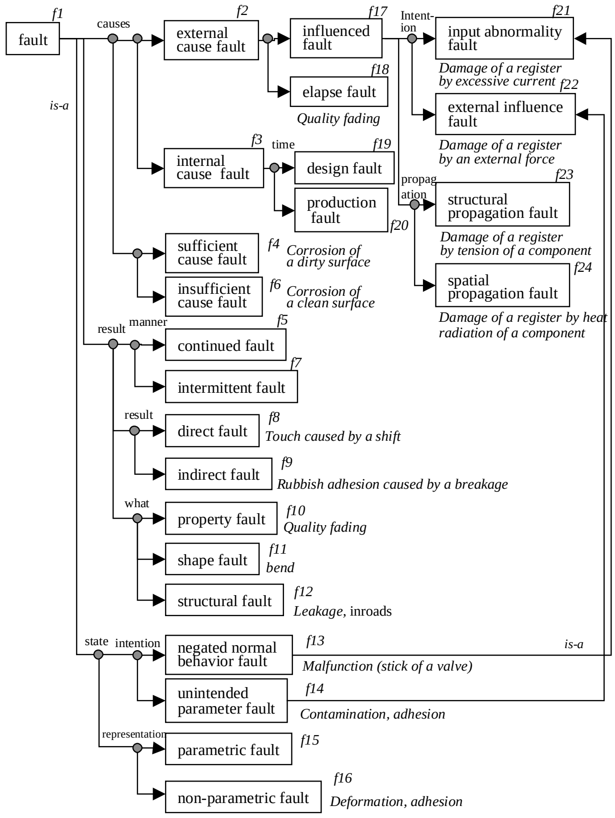


Figure 3: Classes of fault

The fault event model represents all the *fault events*[t43] which are permitted to occur in the object model. A fault event model is composed of *cause of the fault*[t13], action and *fault state*[t17]. The *cause of the fault* and *fault state* are represented as a quadruple: <*subject, attribute, comparator, value*>, respectively, where *subject* takes the values such as *obj*, denoting an *object*, or *env*, denoting an environment. By environment, we mean the neighborhood of the objects of interest and it has attributes such as *temperature, pressure, thing*, and so on. The attribute, *thing*, takes two values, *exist* or *no-exist*.

We have identified 55 fault events some of which are shown in Table 1 where the states with “*” are either ***optional causes***[t16] which are not necessary but facilitate the state change into the faulty state or ***indirect results***[t22] which are not always but sometimes induced associated with the ***direct result***[t21]. By “Actions” we mean the key behavior or action taken to change the state of the object from the normal state to the fault state.

We need another model to enable the system to reason about a portion of *(abnormality) propagation events*[t44] some of which are shown in Table 2. It is mainly for connecting the physical and conceptual parameters and bridging the component model and its environment. The first model says the pressure of the environment is transmitted to the component if they contact each other. These

Label	Causes of fault	Action	Fault states
adhesion	env:thing=exist and obj:phase=solid	obj:adhere (short)	obj:thing=exist, *obj:shape≠normal, *obj:friction-resist>normal
quality fading	*env:temparature≠normal	obj:fade (long)	obj:strength<normal, obj:quality≠normal, *obj:surface≠normal
corrosion	obj:phase=solid and *obj:surface≠normal	obj:corrode (long)	obj:strength<normal, *obj:shape≠normal
collision	env:thing=exist	thing:collide (long)	env:pressure>normal, *obj:surface≠normal
touch	obj:position≠normal	obj:touch (instant)	env:pressure>normal
breakage	(obj:strength<normal or env:pressure>normal) and obj:phase=solid	obj:break (short)	obj:shape≠normal, *env:thing=exist

Table 1: Examples of the fault event model

Label	Causes of fault	Action	Fault states
pressure transmission	env_1:pressure>normal and contact(obj_1, obj_2)	pressure:transmitted (short)	env_2:pressure>normal
movement of a thing in the neighborhood	env_1:thing=exist and near(obj_1,obj_2)	thing:move (short)	env_2:thing=exist
by the liquid flow	env_1:thing=exist and flow(obj_1,obj_2)	thing:flow (short)	env_2:thing=exist
due to the inclusion relation	env_1:thing=exist and include(obj_1,obj_2)	thing:move (short)	env_2:thing=exist
heat conduction to neighborhood	env_1:temprature>normal and near(obj_1,obj_2)	heat:flow (short)	env_2:temprature>normal

Table 2: Examples of abnormally propagation event models.

Variable name	Value A	Value B
1. sufficiency of cause	only *sufficient cause fault*[f4]	*insufficient cause fault*[f6]as well
2. directness of the result	only *direct fault*[f8]	*indirect fault*[f9] as well
3. attributes changed	only *property fault*[f10] and *shape fault*[f11]	*structural fault*[f12] as well
4. kinds of propagation	only *structural propagation fault*[f23]	*spatial propagation fault*[f24] as well
5. elapse or influenced	only *influenced fault*[f17]	*elapse fault*[f18] as well
6. intentional or unintentional	only *negated normal behavior fault*[f13]	*unintended state fault*[f14] as well
7. parametric or non-parametric	only *parametric fault*[f15]	*non-parametric fault*[f16] as well
8. fault time	only *after-fault time*[t30]	*fault process time*[t29]

Table 3: Control variables and their alternative values for use of controlling the search

pieces of propagation event models can deal with *spatial influence*[t41], *external influence*[t39], and so on.

4.2 Concepts for controlling diagnosis

Our aim here is an interactive system in which the human users can control its search space to enumerate possible deeper causes of a given symptom. The information users need is not numerical values, indicating a possibility of "deeper causes", which cannot suggest anything about what cause the system is going to find but categorical information which suggests them "*what type of causes*" the system is looking for.

We identified eight control variables included in the fault ontology for use of controlling the search as shown in Table 3. These control variables represent diagnostic assumptions specifying the scope of diagnosis. Each variable takes two alternative values A and B. When all the control variables are set to the alternative A, the system's capability corresponds to that of the conventional GDE-like systems discussed in Section 3. After running the system in all A mode, if the user want to find "deeper causes", then she/he could change some of the values of control variables and make the system go further.

4.3 Reasoning process and example

We have implemented a reasoning system which integrates two reasoning engines. The one is a qualitative reasoning system using the component model of correct behavior represented by *intended parameters*[t3] and *physical parameters*[t5] (called constraint model) [Kitamura *et al.*, 1996a and 1997]. The another is based on the fault event model discussed thus far [Kitamura *et al.*, 1996b]. In this paper, we concentrate on control of the fault-hypothesis generation process in the diagnostic process due to space limitation. See those papers for more details of the reasoning method.

When all the control variables are set to the alternative A, given a symptom represented by a *physical parameter*[t5], the reasoning engine using the constraint model generates fault-hypotheses representing malfunctions of components and *causes of the symptom*[t42].

When some of the control variables are changed to the alternative B after running the system in all A mode, the hypotheses generated before are converted to the fault event model according to the interpretation knowledge [Kitamura *et al.*, 1996b]. Then, causal chains of *fault events*[t43] are derived by the other reasoning engine using the fault event model and the object model. When *cause of the fault*[t13] identified can be converted to a *physical parameter*[t5] in the constraint model, the deeper causes are derived using the constraint model again.

Let us take the example shown in Figure 1. Given the decrease of the power generated as a symptom, firstly, the reasoning engine using the constraint model may identify the malfunction of the turbine (and other possibilities). Assume that the user wants to find out a deeper cause of it. Then, the decrease of torque of the turbine (*cause of the symptom*[t42] identified) is converted to deformation of the blade of the turbine in the object model[1].

Figure 4 shows an example of the reasoning results given that value. When the variables 6, 7 and 8 are set to B, all nodes other than those which are shaded are inferred. If the suggested causes are found not guilty, then he/she could set the variable 3 to B to find the possibility of *structural fault*[f12]. Then, a cause, "touch of the blade and chamber" is inferred. In the case of a very old turbine, he/she may set the variable 5 to B and then "the quality fading of the blade" is suggested. To go further, if the forth variable is set to B to find the possibility of *spatial propagation fault*[f24], then "the thing might have been come from the super-heater" is obtained.

6. Evaluation and Limitations

Evaluation of the system has been done using two examples: turbine and transformer troubleshooting. Concerning the former, we consulted a textbook of turbine troubles and all the causes listed in the book including deeper ones have been successfully enumerated. The latter is more realistic. We consulted an expert of transformer repair. He had a document in which the all typical troubles are listed. The system was able to enumerate all causes. Needless to say, we need evaluation on another aspect, that is, on how many ridiculous causes were also enumerated as well as plausible and realistic ones. For example, the system enumerated 22 causes in total for a symptom, 3 of which cover all the causes listed in the document, 3 of which the expert accepted as causes of fair possibility, 16 of which he did not reject saying that they are not impossible. There is no ridiculous one.

We aim combination of the generic fault event model and the object model specific to the target system, while the models of faulty modes of components (e.g., shown in [Struss and Dressler, 1989]) are specific to the component. At first glance, the former might look ad hoc. However, its components are based on the ontology so that they are well-founded and the model is designed to be as general as possible. This makes the model reusable across various domains. In the two applications mentioned above, almost all fault events are valid in the two domains except those specific to the electrical domain.

[1] This conversion is made according to the interpretation knowledge specific to the turbine. There is, however, general interpretation knowledge [Kitamura *et al.*, 1996b].

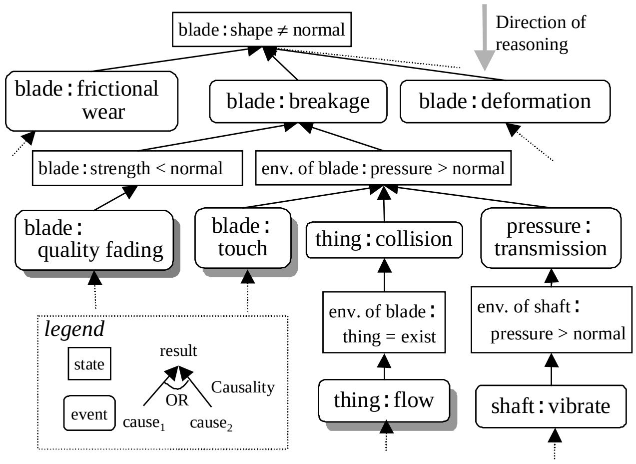


Figure 4: An example of the step-wise reasoning process

It has limitations. The discussion made in this paper could be valid only within the device-oriented modeling. Another ontology based on process-oriented framework needs to be developed. In this paper, we discussed only enumeration of possible fault-hypotheses without the verification and sensing facilities, since we concentrate specification of the search space based on the ontology.

7. Related Work

As discussed in Section 1, the research to date on the capability of diagnostic systems focus mainly on logical aspect. For example, in [Poole, 1989; Console and Torasso, 1991], the differences between logical definition of fault in the constraint-based diagnosis such as [Reiter, 1987] and that in the abductive diagnosis such as [Pople, 1973] are discussed. The logical differences can also explain the difference between the constraint-based diagnosis and our method based on the fault event model which can be viewed as a kind of abductive reasoning. Our point, however, is not the reasoning framework but the characterization of knowledge represented in the framework from the viewpoint of the fault process.

Struss discusses the logical assumptions of a diagnostic process and a framework for "shift of focus of attention and that of suspicion" such as correctness of wire and questioning the observations [Struss, 1992a]. Our classes of faults and the control variables can be viewed as conceptualized categories of such "focus of attention" from the viewpoint of the fault process in order to relax a kind of diagnostic assumptions interactively.

The theoretical frameworks on the integration of the model of correct behavior and the fault model are investigated elsewhere (e.g., [Struss and Dressler, 1989; Console and Torasso, 1990], in general, multiple models [Struss, 1992b]). We simply use the fault event model for searching the deeper causes of the malfunction identified using the model of correct behavior.

Davis investigates four kinds of relaxation of the implicit constraints in diagnosis such as "structural change" and "reverse direction of current flow" [Davis, 1984].

The eight control variables proposed in this paper are an extension of this idea.

An idea of hidden interaction is investigated in [Böttcher, 1995] in order to deal with an unusual interaction between components like “leakage”. Our ontology covers such interactions. General fault mechanisms are discussed in [Purna and Yamaguchi, 1996].

Cascading defects are investigated in [Tatar, 1996], in which it is pointed out that the GDE-like systems sometimes cannot enumerate all cascading defects and hence the undetected defects may cause the already repaired components to break again. We analyzed such a situation (e.g., Figure 1) as the fault process in general and tried to find out such hidden faulty components (the superheater) and the deeper causes of the malfunction.

7. Concluding remarks

We have discussed the ontology of fault including concepts for the fault process and categories of faults, aiming at conceptual categories of a part of diagnostic assumptions. We showed the ontology helps us characterize the existing systems to explicate the capabilities and limitations of them. The reasoning system we developed was successfully evaluated and demonstrated that it could help users enumerate “deeper causes” interactively. Future work includes formalization of the ontology.

References

[Böttcher, 1995] C. Böttcher. No fault in sructure? - how to diagnose hidden interactions, In *Proc. of IJCAI-95*, 1728-1733, 1995

[Console and Torasso, 1990] L. Console and P. Torasso. Integrating models of the correct behavior into abductive diagnosis. In *Proc. of ECAI-90*, 160-166, 1990.

[Console and Torasso, 1991] L. Console and P. Torasso. A spectrum of logical definitions of model-based diagnosis. *Computational Intelligence*, 7(3):133-141, 1991.

[Davis, 1984] R. Davis. Diagnostic reasoning based on structure and behavior, *Artificial Intelligence*, 24: 347-410, 1984.

[de Kleer and Williams, 1987] J. de Kleer and B.C. Williams. Diagnosing multiple faults, *Artificial Intelligence*, 32: 97-130, 1987.

[Hamscher *et al.*, 1992] W. Hamscher et al. (eds.), *Readings in Model-based Diagnosis*. Morgan Kaufmann, 1992.

[Kitamura *et al.*, 1996a] Y. Kitamura et al. 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, 1996.

[Kitamura *et al.*, 1996b] Y. Kitamura et al. Diagnosis based on fault event models. *Working papers of 7th International Workshop on Principles of Diagnosis (DX-96)*, pp.115-122, 1996.

[Kitamura *et al.*, 1997] Y. Kitamura, M. Ikeda, and R. Mizoguchi. A causal time ontology for qualitative reasoning, In *Proc. of the IJCAI-97*, pp.501-506, 1997

[Mars, 1995] N.J.I. Mars (Ed.). *Towards Very Large Knowledge Bases*, IOS Press, 1995

[Mizoguchi and Ikeda, 1997] R. Mizoguchi and M. Ikeda. Towards ontology engineering. In *Proc. of PACES/ SPICIS '97*, 259-266, 1997.

[Poole, 1989] D. Poole. Normality and faults in logic-based diagnosis, *Proc. of IJCAI-89*, 1304-1310, 1989.

[Pople, 1973] H. E. Pople. On the mechanization of abductive logic. In *Proc. of IJCAI-73*, 147-152, 1973.

[Purna and Yamaguchi, 1996] Y. W. Purna and T. Yamaguchi. Generating and testing fault hypotheses with MODEST. In *Proc. the Third World Congress on Expert Systems*, 2:954-961, 1996.

[Reiter, 1987] R. Reiter. A theory of diagnosis from first principles, *Artificial Intelligence*, 32:57-96, 1987.

[Struss and Dressler, 1989] P. Struss and O. Dressler. “Physical negation” - integrating fault models into the General Diagnostic Engine. In *Proc. of IJCAI-89*, 1318-1323, 1989.

[Struss, 1992a] P. Struss. Diagnosis as a process. In [Hamscher et al., 1992], 408-418, 1992.

[Struss, 1992b] P. Strus. What’s in SD? Towards a theory of modeling for diagnosis. In [Hamscher et al., 1992], 419-449, 1992.

[Tatar, 1996] M. M. Tatar. Diagnosis with cascading defects. In *Proc. of ECAI-96*, 511-515, 1996.