Proc. of Tenth International Workshop on Principles of Diagnosis (DX-99), pp.118-128, June 8-11 1999

An Ontological Analysis of Fault Process and Category of Faults

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

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. The capability of a model-based system largely depends on the characteristics of the model used in it. Our aim here is explication of capabilities and limitations of models in model-based diagnostic systems, in particular limitation of the "depth" of the cause of faults identified by a diagnostic mechanism using a kind of model. We focus on the process in which faults are induced and establish an ontology of faults including several categories of faults. It provides us with a vocabulary for specifying the scope of a diagnostic activity performed by a reasoning mechanism using a kind of model. Firstly, the ontology enables us to specify the performances of the models in diagnostic systems, which makes the systems transparent to users who want to know what type of faults the system can diagnose when they describe a target model for a reasoning mechanism. Next, we develop an interactive system to enumerate the "deeper causes" of a malfunction. The ontology enables the human users to control the diagnostic system in a plausible-first way by relaxing diagnostic assumptions interactively

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. The performance of a model-based system composed of a model of a target system and a reasoning mechanism largely depends on the characteristics (such as broadness and completeness) of the model and on its limitation. An assumption made by a reasoning mechanism also limits the performance of the system. 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 the performances of the model-based diagnostic systems (Hamscher *et al.*, 1992) by careful and thorough investigation of the models used in them 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 of the performance mainly of the reasoning mechanism from the logical points of view such as hypothetical reasoning (Poole, 1989; Console and Torasso, 1991) and multiple faults (de Kleer and Williams, 1987; Tatar, 1996). The former research pointed out the differences between the model used in the constraint-based diagnosis and that used in the abductive diagnosis. The former model represents the correct (intended) behavior of the target system, while the latter model represents the abnormal (unintended) phenomena which happen when faults occur. Our aim here is to identify richer vocabulary including such concepts from the viewpoints not of logic but of ontology of physical systems.

Pioneer work is 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 in the model. For example, many of the constraint-based diagnostic systems using the component models representing their correct behavior (e.g., GDE (de Kleer and Williams, 1987)) do not deal with faults caused by topological change among the components. Struss discusses the logical assumptions of diagnosis and a reasoning framework for "shift of focus of attention and that of suspicion" (Struss, 1992a).

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 the next section 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. Thus, the many of the limitations and characteristics of the models are left implicit. In order to specify the limitation of the "depth" of the cause identified by the model-based diagnostic systems, analysis of the causal chains from the initial fault to the symptom, called *fault process*, is needed, while many of the conventional research efforts have focussed mainly on the resultant states of the fault process. Categorization of the phenomena appearing in the fault

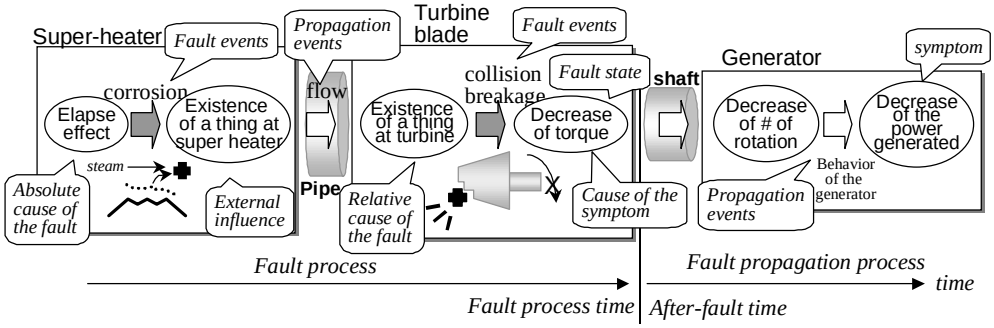


Figure 1: Example of a fault process

process enables us to specify how deep the cause is in the causal chains and categories of the faults identified.

The explicit specification of such conceptualization is called ontology (Gruber 1993). Its roles include to specify the capability and underlying assumptions of knowledge-based systems, which is critical to usability and extensibility of the systems and the successful knowledge reuse (Mizoguchi and Ikeda, 1997).

This paper ontologically analyzes the *fault process*, i.e., causal chains in which the faults and the symptoms are induced, aiming at articulation of concepts which can categorize the fault processes and the faults. The ontological questions to answer include how the observed symptom is induced by the initial fault, what types of phenomena appear in the fault process, and what categories of the deeper causes of faults exist. Consequently, we define concepts related to the fault process such as *fault event* (a category of phenomena) and *external influence* (a category of cause) as part of an ontology of faults. On the basis of these investigations, we identify several categories of faults such as *structural fault* and *spatial propagation fault.*

The ontology of faults provides us with a conceptual vocabulary to explicate the scope of a diagnostic activity performed by a reasoning mechanism using a kind of model. Firstly, it enables us to specify the performances of the models in diagnostic systems, which makes the systems transparent to users who want to know what type of faults the system can diagnose when they describe a target model for a reasoning mechanism. Taking GDE based on the component models representing their correct behavior as an example, we will show it can diagnose rather limited types of faults.

We next develop a fault model and an interactive reasoning method to enumerate the deeper causes of malfunction. The method complements a diagnostic system using the component models representing their correct behavior collaboratively in order to explore deeper causes of the malfunctions identified by the system.

Because this paper aims at an ontology of not the diagnostic task but the fault process, the ontology includes neither task structure of diagnostic task nor diagnostic process. When we design an ontology, in general, we need to discuss the methodology, the content of concepts, the terms, and the axioms (Mizoguchi and Ikeda, 1997). Our research has just finished up to the third step. Axiomatization of the ontology is out of focus in this paper.

Motivation

In this section, we would like to explain our motivation of this ontological analysis of faults using an example from the viewpoint of development of diagnostic systems. Our goal here is to develop a diagnostic system to enumerate the deeper causes of faults. The many conventional systems identify relatively shallow causes of faults such as the location of the faults.

For example, consider a fault in a power plant shown in Figure 1. The story is as follows: The fault has been initiated by the quality fading of the inner pipe of a super-heater due to passing of time (called the elapse effect). Then some pieces of the inner pipe became to exist in the steam flowing 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 rotation speed of the shaft decreased to finally reduce the energy generated by the generator which is observed as a symptom.

A typical constraint-based diagnostic system using the component models representing their correct behavior (e.g., GDE) tries to find a faulty component by reasoning about the parameter values in a resultant state after the fault has occurred. Such a model usually does not include the existence of some pieces, its causes (such as time passing) and its effects (breakage). Thus, in this example, only the turbine blade is identified as the faulty spot explaining the symptom. We can say that it does 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.

Such limitation is not because of the reasoning mechanism but because of the characteristics of the model. If we describe such a fault model that includes the existence of pieces made by the quality fading due to the passing of time and the unintended flow of the pieces, the

reasoning mechanism can generate the deeper causes. The problems with describing such models are the primitives used in the model and their generality. Furthermore, even if we are given such a complete model, it is not realistic to expect a fully automatic diagnostic system for "deeper causes" because of the almost infinite search space.

What we need in order to develop a diagnostic system which can enumerate deeper causes are (1) categorical concepts as primitives for describing fault models, and (2) interactive control of the scope of diagnosis. Our ontology of faults aims at contribution to these issues. For the former issue, we analyze the causal chains of abnormal events such as breakage which occur in the processes to the malfunction of the components. Consequently, we have identified the concepts shown in Figures 1 and 2 which represent such processes called ***fault processes***[t45] (italic represents a concept shown in Figure 2 with own ID number headed by 't'). These concepts are used by the modeler to describe fault models.

For the latter issue, the classes (categories) of faults enable the human users to control the diagnostic system in a plausible-first way by interactively relaxing the diagnostic assumptions in terms of well-defined concepts. We show the classes of faults in the next section and will discuss the stepwise diagnosis using them later.

An Ontology of Faults

Concepts for the Fault Process

A ***state***[t1] represents values of one or more parameters. A parameter representing a state intended by the designer 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. In general, the model of the correct behavior used in the constraint-based diagnosis is in terms of *intended parameters*[t3] and *physical parameter*[t5]. The models used in the abductive diagnosis are mainly in terms of *unintended parameters*[t4].

We consider a *fault process*[t45] is a chain of ***event***s[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 and ***elapse effect***[t32]. The latter represents an event which naturally happens as time passes such as quality fading. Influence is further divided into ***input***[t38] and ***external influence***[t39] according to if it is intended or not. On the other hand, it is also divided into ***structural influence***[t40] and ***spatial influence***[t41] according to the way of influencing. Many component-based diagnostic systems treat the abnormal values of *input*[t38] of the components.

A ***fault event***[t43] is defined as an irreversible change of the state of a component. The state of the *cause*[t7] (it is an

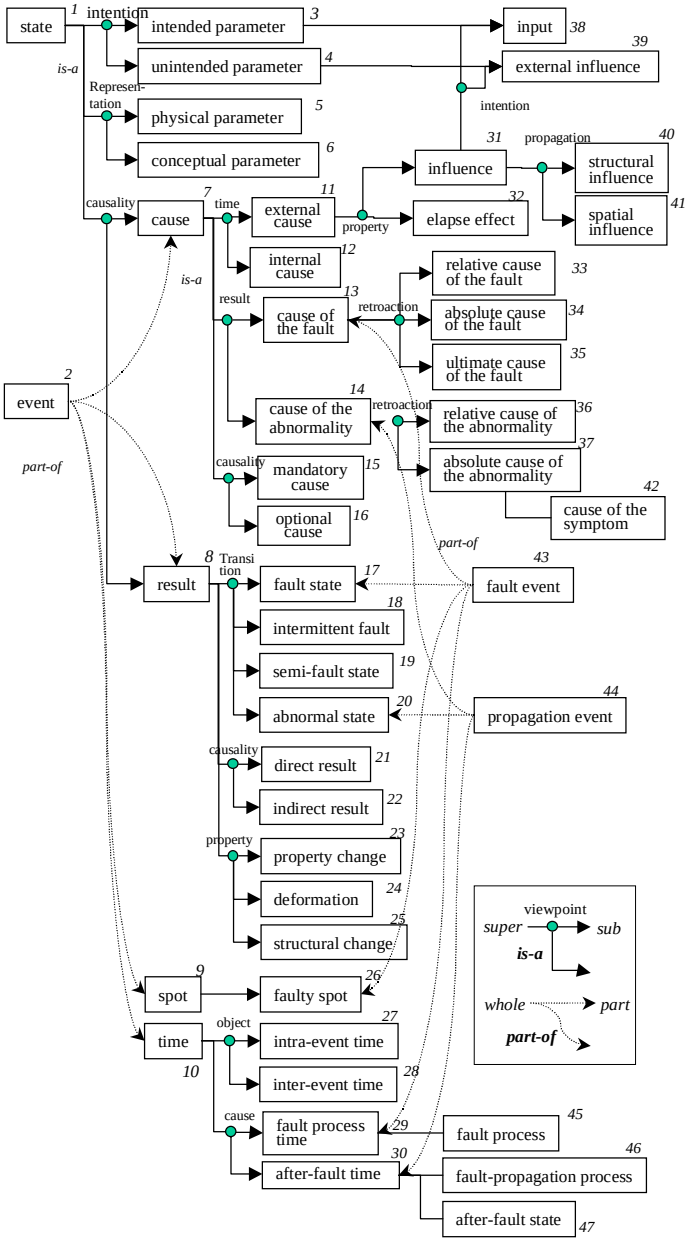


Figure 2: Concepts for the fault process

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 is not changed, we call it ***abnormal state***[t20] caused by ***propagation event***[t44] which means that the output of the component is abnormal only because of the propagation of abnormal input (e.g., "flow" event in the 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 target system which is currently under consideration 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 system under consideration. 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] (e.g., 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.

Classes of Faults

We also identify categories of faults 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]. In general, the fault models represent the latter faults. 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, faults are partitioned into the following three categories; ***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, and ***structural fault***[f12] which represents change of location or that of topological structure (***structural change***[t25]) such as leakage and touch. Although the shape and structure are also kinds of the properties, we distinguish them because they plays a crucial role in the component's functioning and the possibility of deformation is in many cases different from the other properties.

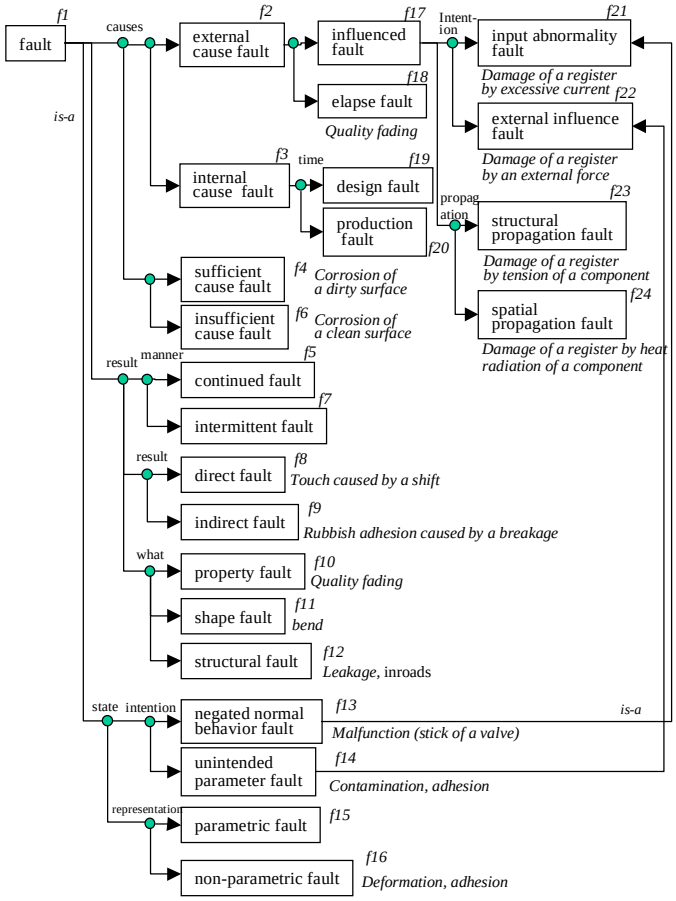


Figure 3: Classes of Faults

Performance of Models in Diagnostic Systems

Let us describe performance of GDE based on component models representing their correct behavior (de Kleer and Williams, 1987). Figure 4 shows the scope which the GDE can diagnose using such a model. Concerning the result of diagnosis, since the model in GDE is composed of the physically structured components and their static and normal behavior, it tries to identify malfunctioning structural 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*

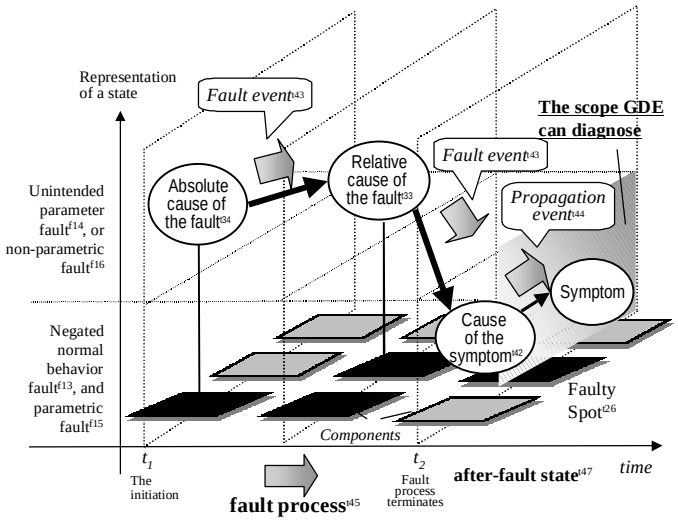


Figure 4: The scope which GDE can diagnose

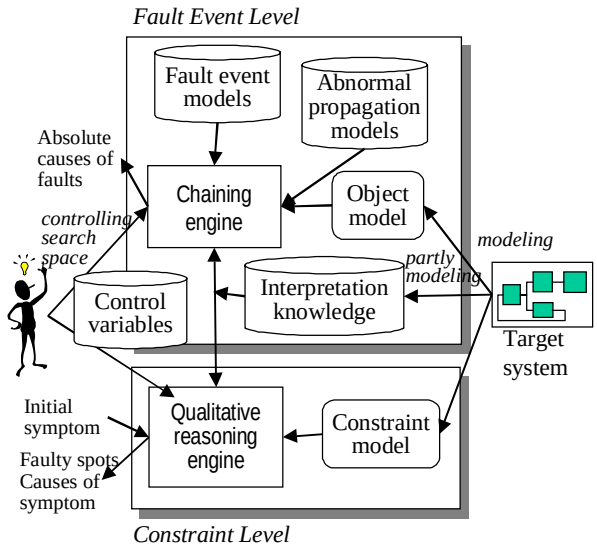


Figure 5: The architecture of our diagnostic system

fault[f14] or *non-parametric fault*[f16] (e.g., corrosion). The former corresponds to its difference with the abductive diagnosis based on the fault model in terms of *unintended parameters*[t4].

Further, GDE based on the device ontology 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].

Development of a Diagnostic System for Deeper Causes

Architecture

The ontology of faults discussed thus far suggests us the necessity of creating a reasoning system and describing a model which can deal with wider scope of faults, that is, one covering so-called "deeper causes", such as *external influence fault*[f22] and *spatial propagation fault*[f24]. 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) introduction of a fault model as specification of the wider search space, (2) integration of the fault model and the constraint model, and (3) development of an interactive system with human experts who can control the direction to pursue with the help of well-conceptualized types of fault. The architecture of our diagnostic system developed is shown in Figure 5.

Firstly, we have described general models of the fault process, which consist of fault event models and abnormal propagation models. A rule-chaining-type reasoning engine generates causal chains of the events which occur in a target system using the general fault event models with an object model specific to the target system.

Secondly, in order to search the causes efficiently, our system also uses the component models representing their correct behavior in terms of *intended parameters*[t3] and *physical parameters*[t5] (called constraint model). The system firstly finds the causes of the given symptom using the constraint model and then searches their deeper causes using the fault model. We have developed a reasoning system which integrates two reasoning engines for the two kinds of models as shown Figure 5. The reasoning using the constraint model shown in (Kitamura and Mizoguchi, 1996a; 1997) is called constraint level. That using the fault model partly discussed in (Kitamura *et al*., 1996b) is called fault event level. The another kind of model called the interpretation knowledge is also needed to bridge the two levels.

Lastly, in order to control the search space of deeper causes, we identified control variables representing "*what type of causes"* which the diagnostic system is looking for. By changing values of the control variables, the human users can control the search space to enumerate possible deeper causes of a symptom.

In this section, we discuss the first two issues. We show the models used in our system and reasoning methods. The last issue will be discussed in the next section.

Object Models (Figure 6)

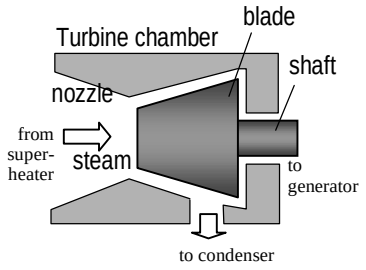
The object model is a model for anything which can be said to be faulty (called *objects*) at the fault event level. The objects include medium such as liquid and gas as well

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 models

Label	Causes of the abnormality	Action	Abnormal state
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 abnormality propagation models.




```
blade:phase=solid,
blade:working=true,
blade:inner-space=false,
chamber:phase=solid,
chamber:working=false,
chamber:inner-space=true,
steam:phase=gas,
steam:working=true,
    ...
```

```
include(chamber, (blade,shaft,nozzle,steam))
contact(blade,shaft),
flow(steam,nozzle,blade),
contact(steam,(blade,nozzle,chamber))
  ....
```

Figure 6: An object model of a steam turbine

as structural components (or devices) such as turbines and pipes. Figure 6 shows an object model of a steam turbine in Figure 1 as an example. The object model consists of description of attributes of the objects (right part in Figure 6) and information of connection among objects (lower part). The connection information includes the structure represented by predicates such as "include" and flow of medium represented by "flow".

Fault Event Models (Table 1)

In order to reason about the deeper causes in *fault process time*[t29], we have developed general models of the *fault process*[t45]. 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.

Abnormality Propagation Models (Table 2)

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 of an object is transmitted to another environment of another object if the objects contact each other. These pieces of propagation event models can deal with *spatial influence*[t41], *external influence*[t39], and so on.

Constraint Models

The constraint model contains qualitative component models representing their intended behaviors. A component model consists of a set of physical parameters,

qualitative constraints over the parameters, ports for connections and causal properties of the parameters. A parameter takes one of the three qualitative values related to the deviation from a normal value. [+]([-]) represents a quantity greater(less) than the normal value, i.e. abnormal values. [0] represents a quantity equal to the normal value. For details of the constraint models, see (Kitamura *et al*., 1996a).

Interpretation Knowledge

The constraint model specific to the target system consists of physical parameters, while the general fault event model contains conceptual parameters. The conversions between them are done according to the interpretation knowledge. The following relations are examples of the interpretation knowledge.

(1) IF humidity = [+] THEN env:water-vapor = exists

(2) IF turbine-efficiency = [-] THEN blade:shape ≠ normal

The first one represents the equivalent relation between the physical parameter "humidity" and the existence of water vapor. This is independent of the component. On the other hand, the second one depends on the turbine and represents a causal relation.

Reasoning Processes

Our diagnostic method consists of two processes, the fault-hypotheses generation and the fault-hypotheses verification. The generation process derives plausible causes of the fault covering the given symptom. The verification process generates all the possible symptoms to be caused by the fault-hypotheses and checks predicted values against actual values. In this article, we concentrate on the fault-hypotheses generation process.

The fault-hypotheses generation process consists of two level reasoning, the constraint level and the fault event level. Given the symptom in terms of *physical parameters*[t5], the qualitative reasoning engine at the constraint level retrospectively propagates abnormal values according to the constraint model. The reasoning results at the constraint level are a set of abnormal parameters whose values have no deeper cause at the level, that is, the absolute causes at the level. Then, the causes are converted to the fault event model according to the interpretation knowledge. At the fault event level, causal chains of fault events are derived on the basis of the fault event models. In the case where causative states at the fault event level can be converted to the constraint level, the deeper causes are derived at the constraint level.

Reasoning using the Constraint Models

The qualitative reasoning engine at the constraint level (Kitamura *et al*., 1996a; Kitamura and Mizoguchi, 1997) is categorized as a type of the reasoning method proposed by de Kleer and Brown (de Kleer and Brown 1984). On the basis of the device ontology, it has two reasoning processes, that is, the intra-component reasoning and the inter-component reasoning. The former propagates abnormal values to the other parameters in the component according to the constraints in the component models. The latter propagates the abnormal values to the neighboring components according to the connection information. It can infer finer-grained causal chains among physical phenomena in transition states and feedback loops on the basis of seven units of time resolution. For more detail, see (Kitamura *et al*., 1996a; Kitamura and Mizoguchi, 1997).

Reasoning using the Fault Event Models

The fault event models make the reasoning on the causal chains of the fault events and the abnormality propagation events. There are two reasoning processes, the retrospective reasoning and the prospective reasoning. The former generates plausible causes which might have caused the given state. The latter generates all resultant states to be caused by the given causative state.

Each reasoning process has two steps, that is, the matching step and the evaluation step. In the retrospective reasoning, the matching step searches for the fault events which have the resultant state matching the current state and the current object model. The fault events found are events which have probably caused the current state and thus they are a part of plausible causal chains for the current state. The causative states of them are the relative causes of the current state. In cases where there are more than one plausible fault events for a current state, the relations among them are OR-relationship. In the evaluation step, the engine views the causative states of the events as new current states. Then, the further matching step is invoked for the new current states in order to detect deeper causes.

According to the fault even model, the changes of the attributes of the objects are identified. On the other hand, the abnormality propagation models enable the reasoning engine to identify the interaction among the objects. The object model restricts the both kinds of events according as the characteristics of the objects and the structural information among them.

In the case of the prospective reasoning, the reasoning is done in the reverse direction. The engine searches for the events that have a causative state which matches the given state. Next, the evaluation step generates new current states according to the description of the resultant state of the events.

There are cases where no event is found in the matching step, because the attributes of the fault events are described in different grain sizes for generality. In such cases, the attributes are generalized according to the hierarchy of the attributes of the fault event. The knowledge representing such hierarchical relations between attributes is called the hierarchical attribute knowledge.

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

Stepwise Diagnosis

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 typical component models discussed the previous sections.

Stepwise Reasoning Process

Normally, the human users firstly set the all variables A then run the reasoning system. 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.

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. 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.

Example of Reasoning

Let us take the example shown in Figure 1. The target component is a steam turbine of a power plant. The turbine is connected to a generator and a super-heater. The symptom given in this example is "the output power of the generator is lower than the normal value". Figure 7 shows the reasoning results of the implemented reasoning engines.

Mode 1: Constraint Level
(All control variables are A)

Firstly, the user runs the diagnostic engine with all control variables having value A. It means the constraint level.

In this mode, firstly, the reasoning engine using the constraint model focuses on the generator whose output is identical with the symptom and generates a relative cause, that is, the revolution of the shaft of the turbine is lower, as well as some possible faults in the generator.

Next, the engine reasons about the turbine. Given the revolution of the shaft is lower, the following causes are generated according to the constraint model of the turbine:

Absolute Cause: turbine-efficiency = [-]
Relative Cause: flow-rate = [-], heat-inflow = [-], inlet-pressure = [-], outlet-pressure = [+]

The relative causes are associated with other components. Then, the abnormal values are propagated to the super-heater which is an upper component.

On the other hand, no deeper cause of the decreases of the turbine efficiency is derived at the constraint level. It means that a *cause of the symptom*[t42] is identified.

Mode 2: Fault Event Level
(Variable 6, 7, and 8 are B)

Assume that the user wants to find out a deeper cause of it. Then, the decrease of the efficiency is converted to deformation of the blade ("blade:shape ≠ normal") of the turbine in the object model using the interpretation knowledge. When the variables 6, 7, and 8 are set to B, all nodes other than those that are shaded shown in Figure 7 are inferred.

First, the engine generates the fault events which have resultant states matching "blade:shape ≠ normal" by consulting the fault event models and the object model. For example, "breakage" is derived as one of the possible fault events. It means that the lower strength of the

component and/or external pressure has caused "breakage" and then the shape becomes abnormal.

Next, the engine views the causative states of the generated events as the resultant states and searches for such fault events that have the resultant state which matches "comp:strength < normal" or "env:pressure > normal". In this example, "transmission" and "collision" are derived.

For the deeper causes of "transmission", "vibrate" is obtained. On the other hand, the causative state of "collision" is an existence of fragments. According to the interpretation knowledge that "IF humidity > normal THEN env:water-vapor = exists", the engine converts the state back to the physical parameter. At the constraint level, the fault that outlet temperature of the super-heater is lower than that of normal is generated.

Mode 3-5: Deeper Fault Event Level (Variable 3, 4, or 5 is B)

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 according to the fault event models and the description of the structure of the turbine in the object model.

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 fourth 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 according to the abnormality propagation models and the description of flow of the steam in the object model, which is the case shown in Figure 1.

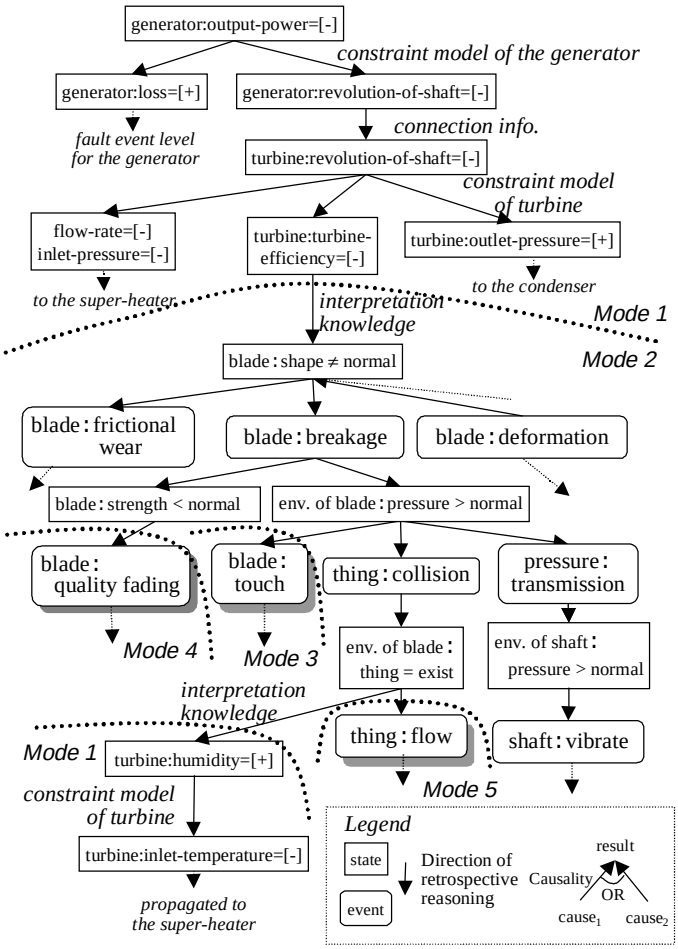


Figure 7: An example of the step-wise reasoning process (generated by reasoning engines)

Evaluation and Limitations

The diagnostic system has been implemented using LISP. 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.

Nevertheless, it has limitations. 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. 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.

Related Work

As discussed in Introduction, the research to date on the capability of diagnostic systems focuses mainly on logical aspect of reasoning mechanisms. 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 also 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 (e.g., the super-heater in Figure 1) and the deeper causes of the malfunction.

In fault-tolerant computing research, categorization of faults has been done (Avizienis, 1982; Kopetz 1982). In (Kopetz, 1982), a bad event is distinguished from a disagreeable state caused by the bad event. They are called "failure" and "fault", respectively. In our terminology, they are called "fault event" and "fault state", respectively.

Avizienis proposes the four-universe model of information systems and classification of faults (called undesired events) which consists of 13 concepts such as internal fault, external fault, permanent fault and transient fault (Avizienis 1982). Our ontology covers 5 in them. His categorization distinguishes the effect by humans (called human-made faults), while ours does not. Our ontology focuses on the process in which the faults are induced, and then has richer categories of causes from the viewpoint of the physical mechanism of faults such as the propagation of the influences of faults.

In the research area on reliability of artifacts and failure physics, categorization of abnormal phenomena is done (Shiomi, 77). We described the fault event models, consulting such literature. The fault event model is a class of concrete fault events which happen in the target system. As a meta-model of the model of the target system, the ontology of faults might include the fault event models. In general, it is hard to partition a set of concepts into an ontology and a model based on the ontology. At this moment, our ontology includes *not* specific phenomena *but* general categories according to general characteristics of attributes or temporal aspects of the events.

Concluding Remarks

We have discussed the ontology of faults 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 models used in the diagnostic 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.

In this paper, we concentrate enumeration of possible fault-hypotheses with interactive control of the search space. An investigation on verification of fault-hypotheses and detection of symptoms remains as future work. In this paper, our ontology is not shown in formal languages. The formalization of the ontology also remains as future work.

Acknowledgments

The authors would like to thank Toshihito Nishihara and Masahiko Ueda for their contributions to this work. The authors are grateful to Mitsuru Ikeda and Takashi Washio for their valuable comments. The authors' thanks also go to the anonymous reviewers for their valuable comments.

This research is supported in part by the Japan Society for the Promotion of Science (JSPS-RFTF97P00701). A part of this paper was prepared under a Reentrustment Contract with the Laboratories of Image Information Science and Technology (LIST) from the New Energy and Industrial Technology Development Organization (NEDO), concerning the Human Media Technology program under the Industrial Science and Technology Frontier Program (ISTF) of the Ministry of International Trade and Industry (MITI) of Japan.

References

Avizienis, A. 1982. The four-universe information system model for the study of fault-tolerance, In *Proc. of 12th*

International Symposium on Fault-Tolerant Computing (FTCS-12), 6-13.

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

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

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

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

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

Gruber, T. R. 1993. A translation approach to portable ontology specifications, *Knowledge Acquisition*, 5(2):199-220.

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

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

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

Kopetz, H. 1982. The failure fault (ff) model, In *Proc. of 12th International Symposium on Fault-Tolerant Computing (FTCS-12)*, 14-17.

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

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

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

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

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

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

Shiomi, H. 1977. *Failure Analysis and Diagnosis*, JUSE Press (In Japanese).

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

Struss, P. 1992a. Diagnosis as a process. In (Hamscher *et al.*, 1992), 408-418.

Struss, P. 1992b. What's in SD? Towards a theory of modeling for diagnosis. In (Hamscher *et al.*, 1992), 419-449.

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