Integrating Model-Based Monitoring and Diagnosis of Complex Dynamic Systems

Franz Lackinger and Wolfgang Nejdl
Christian Doppler Laboratory for Expert Systems
Technical University of Vienna,Austria
e-mail: lackinger@vexpert.dbai.tuwien.ac.at

Abstract

We present a new approach to model-based monitoring and diagnosis of dynamic systems. The presented DIAMON algorithm[1] uses hierarchical models to monitor and diagnose dynamic systems. DIAMON is based on the integration of teleological parameter-based monitoring models and repair-oriented device-based diagnosis models. It combines consistency-based diagnosis with model-based monitoring and uses an extension of the QSIM-language for the representation of qualitative system models. Furthermore, DIAMON is able to detect and localize a broad range of non-permanent faults and thus extends traditional diagnosis which exclusively deals with permanent faulty behavior. The operation of DIAMON will be demonstrated on a real-world example in a multiple-faults scenario.

1 Introduction

Only a few model-based approaches trying to monitor and/or diagnose dynamic systems have been published in the past (e.g. [Dvorak and Kuipers, 1989; Ng, 1990]). Most of them resort to qualitative simulation as an inference engine during the troubleshooting process to predict possible behavior patterns. Additionally, the architecture of the used qualitative system models is either non-hierarchical or contains discrete layers of abstraction.

Unfortunately, all approaches concentrate either on monitoring (i.e. fault detection) or on diagnosis (i.e. fault localization) and therefore miss important aspects of effective and efficient troubleshooting. We present an integrated algorithm which performs both tasks and which can therefore be used as a widely applicable general framework for troubleshooting dynamic systems.

We start in Section 2 with a discussion of the various problems that occur in the course of troubleshooting dynamic systems. In Section 3 we present our approach how to solve these problems and describe the important properties of DIAMON. The operation of DIAMON is shown on a real-world control problem - troubleshooting a central heating system. Section 4 discusses related research work and compares it with our approach, closing with some comments on future work.

[1] DIAMON means DIAgnosis and MONitoring Algorithm

2 Troubleshooting Dynamic Systems

In order to build an integrated control system doing both monitoring and diagnosis in dynamic environments, we have to be aware of the issues which are involved in such a task.

2.1 Dynamic vs. Static Diagnosis/Monitoring

While diagnosis of a faulty component in a static system is a comparatively straightforward sequential process of fault detection, measurement selection and fault localization (e.g. [de Kleer and Williams, 1989]), this process grows more complex in a dynamic system.

Example 1 [Central Heating System:] Consider the central heating system depicted in figure 1. The function of the central heating system is to guarantee that the actual room temperature T_{Room} equals the intended room temperature T_{wanted}.

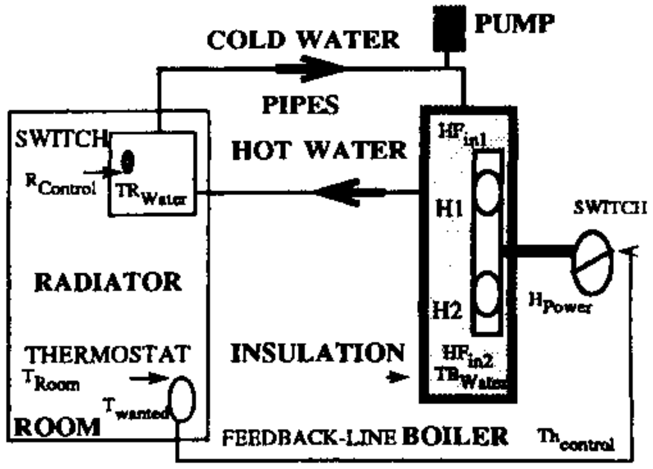

Figure 1: Central Heating System

In the course of monitoring and diagnosing this dynamic system, we have to solve the following problems:

Multiple Layers of Detail: Due to the use of several layers of models (monitoring models, hierarchical diagnosis models), we must take a changing system description and component set into account. In order to improve efficiency, real-time monitoring tends to minimize the amount of parameters under observation, and constraints are usually not component-oriented. In our example, checking the thermostat and the room temperature is enough for most monitoring purposes. Hierarchical diagnosis models use mostly component-oriented constraints at different abstraction levels. In our example, the most detailed level includes all the components depicted in figure 1.

Temporal Observations: Observations consist of sets of measurements at different time points, determining the value of a specified set of parameters at different times. As a consequence, taking measurements sequentially is not feasible in a dynamic system as all parameters in a measurement set have to refer to the same system state (which changes over time). Additionally, the observation rate has to be chosen such that all system states can be observed.

Time Dependent Behavior: We have to take various modes of correct behavior and various system states into account. For example, after turning the thermostat to a higher temperature, the temperature difference between room and thermostat temperature is perfectly normal. However, if they are not equal after a sufficient time period has elapsed, this difference leads to a conflict set.

Intcrmittency of Faults: Static diagnosis ([Reiter, 1987]) assumes that the mode *(ok* or *ab)* of a component does not change during the diagnosis process *(non-intermittency of component modes).* However, this may or may not be recognized in the different situations which are characterized by the parameter sets. A dynamic diagnosis system has to handle this *intcrmittency of faults.* For example, if one of the heating elements is not working, we detect this fault only after a sufficient time period has passed and the room is still not warm enough.

Intra State Consistency: Some diagnosis engines ([Davis, 1984]) use multiple test vectors (parameter sets) which are assumed to be independent and describe only one system state. In dynamic systems we have to describe the system in several system states. Therefore not only conflicts of a parameter set with respect to the correct system state are relevant, but also conflicts describing faulty transitions between these system states have to be included into diagnosis. In our example, assume two system states *heat* and *cool.* The transition from *heat* to *cool* takes place, if $T_{room} > T_{high}$, while the system changes from *cool* to *heat,* if $T_{room} < T_{low}$. If such a transition takes place too early (i.e., if the heater turns off too quickly), we get a conflict with the specified behavior.

The above discussion shows that the current state-of-the-art of model-based diagnosis which is limited to static systems and permanently faulty behavior is unable to solve the discussed problems. It is the goal of our approach to extend model-based diagnosis with respect to dynamic systems and to solve most of the above mentioned problems.

2.1.1 Monitoring - a Necessity in Dynamic Systems

Having designed a structural and behavioral model of a static device we can unambiguously derive predictions about its correct behavior. Therefore, we do not have to monitor a static system as its behavior is precisely determined. However, making predictions about the correct behavior of dynamic systems is a more complicated task.

If we resort to qualitative modeling techniques and use qualitative simulation for behavior prediction, we can usually predict more than one possible behavior pattern. Consequently this requires a monitoring phase in the troubleshooting process to detect inconsistencies between sets of observations and possible correct behavior patterns.

On the other hand, we claim that it is not necessary to observe all possible correct behavior patterns. It is sufficient to determine whether the device fulfills its function or not. For example, we will use a car long after some small deviations from its original behavior specification have occurred. In general this function of a device can be modeled according to its purpose *TEL* (which, for example, is expressed in first order sentences) by a teleological monitoring model (denoted with *TELM)* following the condition

$$TEL_M \wedge \bigwedge_{c \in COMP_{TEL_M}} ok(c) \models TEL.$$

Note that in the above definition $COMP_{TEL_M}$ will be the dynamic system itself (i.e. we can use the abbreviation *ok (device)).* However, we can easily extend this general concept to allow subpurposes of subparts of the system.

In contrast to [Franke, 1989] who proposes the derivation of teleological models from an envisionment, we derive such models heuristically. We claim that a teleologic model contains meta-knowledge about the purpose of the device which can only be determined by a human expert and which includes such knowledge as which faults can still be tolerated.

While monitoring, we have to check only the accordance of the actual behavior with this teleological model under the assumption that the device works correctly.

Definition 1 (Monitoring) *Monitoring* is the process of testing whether

$$TEL_M \wedge ok(device) \wedge OBS_M$$

is consistent.

Usually fault detection alone is not sufficient and we need an additional diagnosis phase to provide sufficient information about fault localization for repair.

Additionally, we have to clarify how the choice of a qualitative model influences the monitoring and diagnosis process.

2.2 Different Types of Qualitative Models

2.2.1 Simulation Models

Doing simulation we are mainly interested in the behavior of the dynamic system over time. This behavior is optimally deduced from a structural model which uses time-varying parameters as its model primitives and which does not necessarily have to contain component-oriented knowledge.

Further we want to derive *all* behavior patterns with respect to the qualitative model no matter whether they describe faulty or correct behavior. Note that the term *all* refers to the chosen level of qualitative abstraction in the simulation model. The partial inclusion of models which describe physically impossible behavior is caused by the qualitative nature of the representation language and the problem-solving strategy of the simulation algorithm ([Kuipers, 1985]).

2.2.2 Monitoring Models

We claim that the main task of monitoring is to check in real time if the purpose of the system is still fulfilled. A monitoring system observes under real-time constraints a small set of parameters and determines their correctness with respect to the system description.

Therefore, a monitoring model usually contains only a few parameters and constraints necessary to describe the teleological purpose of the device. Informally, we restrict the set of simulation models by adding teleology in accordance with the previous discussion about teleologic monitoring models.

Note that a monitoring model can be viewed as an instantiation of the system's teleology. In general, however, system teleology may cover a *broader range of purpose* than the monitoring model actually expresses.

2.2.3 Diagnosis Models

Simulation and monitoring models are both built using parameters and constraints. This parameter-oriented view, however, is not sufficient for diagnosis. Parameters can represent fault symptoms (e.g. the exceeding of thresholds), but faults are usually related to physical components. We assume here that the main aspect of diagnosis is *to provide information for repair*, and only mechanical devices (and not parameters) can be repaired [2]. We therefore have to relate purely parameter-oriented constraints (e.g. QSIM-constraints) to the device components that we want to diagnose and to repair.

Diagnosis models include component-connection information and are typically hierarchically structured. It is obvious that the lowest-level model has to contain only components which can be repaired, also called *smallest replaceable units* or *SRUs*. The choice of the SRUs depends on the availability of a repair-technician as well as on cost considerations.

Diagnosis greatly benefits from hierarchical models. Complex real-world technical systems include intrinsic hierarchies which are related to the modularity or the purpose of the mechanism. Note that not only a device-oriented structural model can be hierarchical; we use hierarchical, parameter-oriented structural models as well to represent decomposable physical quantities (e.g. fluid flow).

We have of course described extreme positions for these models. Sometimes it might be useful to express sub-purposes of sub-mechanisms. On the other hand, diagnosis coupled with repair is also often oriented towards reconstruction of a teleology which may be different from the one used for monitoring ([Friedrich *et al*., 1990]).

The above discussion shows that diagnosis models are more restricted than simulation models [3] but cover a broader range of expertise than monitoring models do.

The adjustment of parameters is a struggle against fault symptoms, not against fault causes.

3 The DIAMON System

The following section introduces the DIAMON system, a qualitative reasoner for model-based diagnosis and monitoring, which incorporates the principles discussed in the previous section.

3.1 Example: Central Heating System

Figure 2 shows the constraint-network of the central heating system: the parameter model and the device model are connected by constraints. The device-model contains three hierarchies consisting of 1 component (Central Heating), 3 components (Room, Boiler, Pipe-System) and 8 components (Thermostat, Radiator, Heater 1, Heater2, Switch, Insulation, Pipes, Pump).

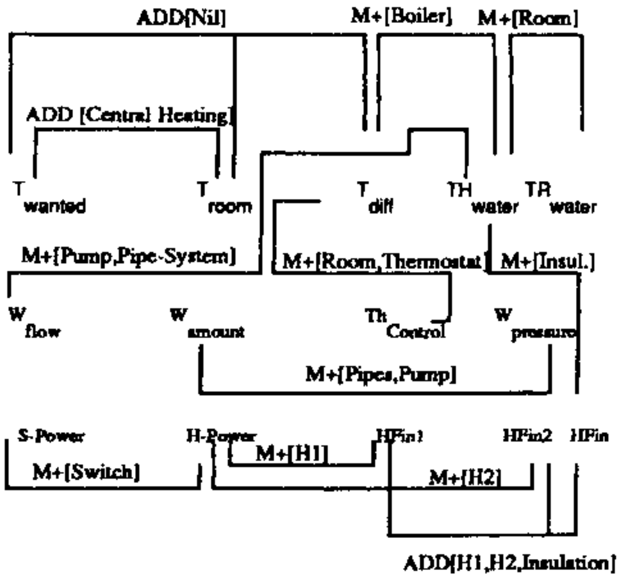

Figure 2: Constraint Network for the Central Heating

We use a flat monitoring model and a hierarchical diagnosis model which consists of two levels. The monitoring level is modeled with 2 parameters and 1 constraint. The first diagnosis level (describing the 3 device-components) uses 7 parameters and 5 constraints, the

[3]with respect to their qualitative expressiveness

SRU-level (describing the 8 device-components) which is used for repair consists of 16 parameters and 13 constraints.

Table 1 explains those parameters of our model that are used for monitoring and diagnosis.

PARAMETER:	EXPLANATION:
T_{wanted}	intended room-temperature (thermostat)
T_{Room}	actual room-temperature
TB_{Water}	actual water-temperature in the boiler
TR_{Water}	actual water-temperature in the radiator
W_{flow}	water-flow in the pipe-system
$Th_{Control}$	control signal from the thermostat (feedback)
$R_{Control}$	signal for the radiator-switch (exogenous)
$Switch$	signal for the water-heater switch
H_{Power}	Heating Power (total)
HF_{in1}	Heating Power from Heater 1
HF_{in2}	Heating Power from Heater 2

Table 1: Model Parameter

3.2 Qualitative Modeling using Device Constraints

We extend the QSIM-language of [Kuipers, 1986] by adding device-oriented knowledge to the constraints.

Definition 2 (Device Constraint) The syntax of a *device constraint* is grammatically defined as follows

$$Dev\_Constr := ((constr)(l\_of\_comp)(corresp\_values)^{+})$$

where *(constr)* and *(corresp.values)* refer to the traditional QSIM definitions of *(constraint)* and *(corresponding — values),* and *(l_of_comp)* denotes the list of device components which are associated by the constraint.

Example 2 The device constraint

$$((M^{+}(Pa\ Pa))(Comp1\ Comp2)((lm1\ lm2)(lm3\ lm4)))$$

is interpreted as follows:

- parameter PB is a rnonotonic function of parameter PA
- the constrained parameters are associated to the device-components C0MP1 and C0MP2
- if the qualitative value of PA is landmark value 1ml (or lm3), then PB simultaneously has the qualitative value lm2 (or lm4).

According to the introduced device-oriented extension, we distinguish between four classes of device constraints in our modeling language.

Teleologic Device Constraints: A teleologic device constraint contains parameters that are used during the monitoring cycle to detect a first fault symptom. It is obvious that the device itself (at its highest level of abstraction) is the device-oriented part in such a device constraint.

Example 3 In our example, the following teleologic device constraint states the desired relationship between the intended and the observed room temperature:

$$((EQUAL(T_{wanted}, T_{Room}))(Central\_Heating)$$
$$((cold\ cold)(t_{room}\ t_{room})(hot\ hot)(inf\ inf)))$$

Tautologic Device Constraints: These constraints exclusively denote relations between parameters that cannot cause inconsistencies due to a component failure. We use them to denote laws of nature, for example energy-conservation, or to relate computation parameters. In contrast to the other types of constraints a tautologic device constraint is not associated to any component as it can not provide any diagnostic information for troubleshooting.

Example 4 In our example, we use a tautologic device constraint to relate the intended room temperature, the actual room temperature and the temperature difference (which is a computation parameter). NIL denotes the empty component set.

$$((ADD(T_{diff}, T_{Room}, T_{wanted}))(NIL)$$
$$((0\ cold\ cold)(0\ t_{room}\ t_{room})(0\ hot\ hot))))$$

Singleton Device Constraints: Some device constraints are explicitly associated to only one component. They express restrictions on internal operations of the component which do not affect its surroundings. An inconsistency between such a device constraint and the observed values is therefore explained by the assumption that the component is behaving abnormally.

Example 5 A singleton device constraint is associated to the radiator, that expresses a qualitative state of the radiator-signal:

$$((EQUAL(R_{Control}, ON))(RADIATOR)(on\ on))$$

Set Device Constraints: This class of device constraints associates at least two components. Although the included components are usually situated at the same level of abstraction, some set device constraints might associate components at different levels of abstractions as well.

Example 6 The rnonotonic functional relation (expressed by an M^{+} — *constraint* (see [Kuipers, 1986])) between the amount of water in the pipe-system and the pressure of water associates the pipes and the pump:

$$((M^{+}(W_{Pressure}, W_{Amount}))(PIPES, PUMP)$$
$$((0\ 0)(p_{normal}\ full)))$$

In this case, we use a set device constraint to avoid the introduction of additional parameters which are not observable. Additionally, set device constraints represent the behavior of connected groups of SRUs (we cannot, replace an *abstract* component like PIPE-SYSTEM).

3.2.1 Layered Modeling Architecture

The modeling architecture of DIAMON consists of three layers:

Monitoring Layer: The monitoring layer of DIAMON contains a teleologic device constraint model which is used to check for faults in the system. All parameters in the monitoring layer are continuously observed to guarantee fast fault detection. Note that if the monitoring layer contains enough parameters to pinpoint the failure of a specific part of the device, then this focusing is done on the first diagnosis level. In this case the first diagnosis level could actually consist of the same parameters as the monitoring level, but additionally include device components.

Diagnosis Layer: The diagnosis layer of DIAMON contains hierarchically structured device constraints which are used for the dynamic refinement of fault localization. It is obvious that the complexity of a device is represented in the hierarchical architecture of the diagnosis layer.

Repair Layer: The repair layer of DIAMON contains device constraints which exclusively relate SRUs. If the diagnosis process reaches the repair layer (i.e. all actual diagnoses only contain SRUs), it switches control to the repair process.

3.3 Algorithm

The following section introduces the algorithmic principles of DIAMON.

3.3.1 The Extended HS-DAG-algorithm

Previous approaches to monitoring have usually concentrated on the control of predefined thresholds. Conversely we view both monitoring and diagnosis from the consistency-preserving point of view which allows us to cover a broader range of detectable faults. The limits of detectability are due to the expressiveness of the underlying constraint language, not to the monitoring/diagnosis algorithm itself.

DIAMON is built on top of the *HS-DA* (algorithm [Greiner *et* a/., 198990] which is an improved version of the Re iter-algorithm [Reiter, 1987] for model-based diagnosis. The device constraints are checked for consistency by a constraint-propagator. If an inconsistency in a constraint is detected, DIAMON adds the associated components to the conflict set.

We have extended the basic *HS-DAG*-algorithm to deal with dynamic systems similarly to the algorithm presented in [Ng, 1990].

3.3.2 Dynamic Model Zooming

We have developed a continuous strategy of dynamic model zooming for the diagnosis layer. *Zooming* denotes the process of focusing the diagnosis process to the relevant parts of the model. In our current implementation we use a *breadth-first* zooming strategy which recursively zooms in the hierarchically deeper level for *all* components which are part of a diagnosis. This strategy guarantees a diagnostic process which is optimal w.r.t. the amount of detectable faults.

We plan to develop an integration of heuristic knowledge and a *best-first* zooming strategy if no safety-criteria are violated.

3.4 The Monitoring/Diagnosis Cycle

Continuing the above discussion we present a formal definition of the DIAMON-algorithm.

Let *M* be the system description consisting of a sequence of qualitative models $m_0, m_1, ..., m_n$, Σ denotes a set of diagnoses Δ (following [Reiter, 1987]).

According to our hierarchical modeling concepts, m_0 is the monitoring layer, m_n denotes the repair layer and mjt *with* $0 < k < n$ are the models of the diagnosis layer. $^{m}actuai \in M$ denotes the current working model used for troubleshooting.

Let P_{m_i} denote the set of parameters of model m_i and let C_{m_i} denote the set of components of model m_i for $0 \leq i \leq n$, $m_i = P_{m_i} \cup C_{m_i}$.

Let further *succ* be the binary relation $M \times (C_M \times C_M) \rightarrow M$ with $succ(m_i, \Sigma) = m_{i+1}$ for i = 0 to n-1. In our current implementation, *succ* is realized by a ***breadth-first*** zooming strategy which refines all device-components of a dynamic diagnosis.

Let further D_s denote a sequence of sets of measurements $d_0, d_1, ..., d_s$ with $cardinality(d_i) \leq cardinality(d_j)$ and $d_i \subseteq d_j$ for $j \geq i$. $d_{actual} \in D_s$ is the current working set of measurements.

Finally let $follow$ be the unary relation $D_s \rightarrow D_s$ with $follow(d_i) = d_{i+1}$ for i = 0 to n-1. DIAMON's implemented $follow$-relation extends $P_{m_{actual}}$ by $P_{succ(m_{actual})} = P_{m_{actual}} \cup follow(d_{actual})$.

In the following description, $CONSISTENT(m, d)$ denotes the call of the theorem prover which checks whether model m and measurement set d are consistent. The result of $CONSISTENT(m, d)$ is either the empty set or the set of conflicts con. $HSDAG(con)$ denotes the call of the HS-DAG algorithm which returns the set of diagnoses (i.e. the minimal hitting sets for con) and $READ(d)$ describes the observation process of parameter set d.

Accordingly the DIAMON-algorithm consists of four steps:

```
1 INITIALIZATION:
   $m_{actual} := m_0;$
   $d_{actual} := d_0;$
   $\Sigma := \emptyset;$
2. MONITORING:
   WHILE ({con .= CONSISTENT($m_{nctualt} d_{actna}$)) = 0)
   DO READ($d_{aciual}$),
3 DIAGNOSIS:
   REPEAT
         $\Sigma := HSDAG(con);$
         $m_{actual} := succ(m_{actual}, \Sigma);$
         $d_{actual} := follow(d_{actual});$
         $READ(d_{actual});$
         $con := CONSISTENT(m_{actual}, d_{actual});$
   UNTIL $(\Sigma \subset m_n)$
4 (REPAIR:
   REANIMATE; GOTO 1)
```

In this paper we concentrate on monitoring and diagnosis and do not discuss repair strategies. We are currently investigating and evaluating various strategies how to restore teleology in a dynamic system doing repair (see also [Friedrich *et* a/., 1990]).

3.5 Example: A Fault Scenario in the Central Heating System

Applying DIAMON to the on-line control of the central heating system yields the following results for the detection and localization of a double fault.

Assume that the radiator is switched off and one heating element (H2) is permanently faulty DIAMON correctly detects that T_{Room} differs from T_{wanted} and switches to the first diagnosis level. Additional parameters and constraints are zoomed in (state s2), i.e. the parameters are measured and the constraints are evaluated. This is done again in state s3 to reach the SRU-level. The diagnosis process finally localizes the double fault *[RADIATOR,H2).*

Table 2 summarizes the control process (sO - s3 denote the qualitative system states (see IKuipers, 1986])).

	$s0$	$s1$
T_{wanted}	T_{Room} std	T_{Room} std
T_{Room}	T_{Room} std	$(Cold\ T_{Room})$ dec
Layer:	Monitoring	Monitoring
Actual Diagnoses	nil	[Central-Heating]

	$s2$	$s3$
T_{wanted}	T_{Room} std	T_{Room} std
T_{Room}	$Cold$ std	$Cold$ std
TB_{Water}	$(Cold\ HOT)$ dec	$(Cold\ HOT)$ dec
TR_{Water}	$Cold$ std	$Cold$ std
$Th_{Control}$	ON std	ON std
$R_{Control}$		0 std
$Switch$		$CLOSED$ std
H_{Power}		ON nil
HF_{in1}		ON std
HF_{in2}		0 std
Layer:	Diagnosis	Repair
Actual Diagnoses	[ROOM], [BOILER]	[RADIATOR,H2]

Table 2: Double-Fault-Diagnosis

4 Conclusion and Related Work

We have presented a new approach to model-based troubleshooting dynamic systems which uses an extended constraint language for hierarchical system models and which is based on consistency-preserving algorithmic concepts for monitoring and diagnosis.

Our work is closely related to that of [Dvorak and Kuipers, 1989] and [Ng, 1990]. The DIAMON system differs from these approaches in some important ways.

First, we use a combined model-based algorithm which integrates both monitoring and diagnosis and which additionally uses hierarchical models and a dynamic zooming strategy.

Second, DIAMON's fault coverage is more complete than those of [Dvorak and Kuipers, 1989] and [Ng, 1990]. In [Dvorak and Kuipers, 1989J faults can be missed due to the use of pre-simulated fault models which do not allow the detection and localization of unanticipated faults. The algorithm of [Ng, 1990] can miss faults if the heuristically chosen incomplete sets of measurable parameters do not represent the actual fault scenario.

The use of a hierarchical modeling architecture for troubleshooting is closely related to to the work of [Hamscher, 1991] who distinguishes between a functional and a physical (repair) model for diagnosis of digital circuits.

We differ from [Dvorak and Kuipers, 1989] in that we do not use fault models and inductively derived fault hypotheses for monitoring. Rather, we rely on the behavior discrepancies to the correct behavior model, which allows us to handle unanticipated faults.

Acknowledgements

We especially thank Benjamin Kuipers who provided the Common LISP-implementation of QSIM (Q2). Brian Falkenhainer and anonymous referees have provided valuable comments on an earlier draft of this paper. Alois Haselbock implemented most of the DIAMON system as part of his M.S. Thesis [Lackinger and Haselbock, 1991].

References

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

[de Kleer and Williams, 1989] Johan de Kleer and Brian C. Williams. Diagnosis with behavioral modes. In *Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI),* pages 1324-1330, Detroit, August 1989. Morgan Kaufmann Publishers, Inc.

[Dvorak and Kuipers, 1989] Daniel Dvorak and Benjamin Kuipers. Model-based monitoring of dynamic systems. In *Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI),* pages 1238-1243, Detroit, August 1989. Morgan Kaufmann Publishers, Inc.

[Franke, 1989] David W. Franke. Representing and acquiring teleological descriptions. In *Proceedings of the Workshop on Model Based Reasoning,* pages 62-67, Detroit, 1989.

[Friedrich *et al,* 1990] Gerhard Friedrich, Georg Gottlob, and Wolfgang Nejdl. Towards a theory of repair. Technical report, Technical University of Vienna, November 1990. Submitted for publication.

[Greiner *et al,* 198990] Russell Greiner, Barbara A. Smith, and Ralph W. Wilkerson. A correction to the algorithm in Reiter's theory of diagnosis. *Artificial Intelligence,* 41(l):79-88, 1989/90.

[Hamscher, 1991] Walter Hamscher. Modeling digital circuits for troubleshooting. *Artificial Intelligence,* 1991. to appear.

[Kuipers, 1985] Benjamin Kuipers. The limits of qualitative simulation. In *Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI),* pages 129-136, 1985.

[Kuipers, 1986] Benjamin Kuipers. Qualitative simulation. *Artificial Intelligence,* 29:289-388, 1986.

[Lackinger and Haselbock, 1991] Franz Lackinger and Alois Haselbock. Qualitative models for simulation and control of dynamic systems. In *AISB91,* Leeds, U.K., April 1991. Springer-Verlag. to be published.

[Ng, 1990] Hwee Tou Ng. Model-based, multiple fault diagnosis of time-varying, continuous physical systems. In *Proceedings of the IEEE Conference on Artificial Intelligence Applications (CAIA),* pages 9-15. IEEE Computer Society Press, 1990.

[Reiter, 1987] Raymond Reiter. A theory of diagnosis from first principles. *Artificial Intelligence,* 32:57-95, 1987.