In a book with the curious title *Normal Accidents*, Charles Perrow examines several of the most notable accidents involving complex systems in the modern industrial world—accidents such as the 1979 Three Mile Island nuclear power accident, the 1977 New York City blackout, and the 1969 Texas City explosion of a butadiene refining unit. Perrow highlights the difficult job of plant operators who are responsible for physical systems with complex interactions and tight coupling. With current monitoring technology, alarms are triggered whenever fixed thresholds are exceeded. A nuclear power plant, for example, can have over a thousand distinct alarms, and hundreds of them can be activated within a minute, as in a loss-of-coolant accident. In such situations, process operators tend to overlook relevant information, respond too slowly, and panic when the rate of information flow is too great.

Not surprisingly, operator advisory systems have become an important area of application for expert systems. ESCORT (an expert system for complex operations in real time) and REALM (a reactor emergency action level monitor) are two of many expert systems developed for process industries. These systems aim to reduce the cognitive load on operators, usually by helping to diagnose the cause of alarms and possibly by suggesting corrective actions. Most of these expert systems get their knowledge of symptoms, faults, and corrective actions through the usual process of codifying human expertise in rules or decision trees. The problem, as with all expert systems, is reliability. As Denning observes:

> The trial-and-error process by which knowledge is elicited, programmed, and tested is likely to produce inconsistent and incomplete databases; hence, an expert system may exhibit important gaps in knowledge at unexpected times.

Obviously, these “gaps in knowledge” can have serious consequences in some process industries.

An alternate approach—one that is not based on an expert’s imperfect recall of symptoms and faults—is to use a model of the process to predict its behavior or to check consistency among observed variables. When observations disagree with the model’s predictions, some diagnostic technique is initiated to identify the fault candidates. These model-based approaches to diagnosis have emerged from two different communities. In the engineering community, fault detection and isolation techniques generally rely on a precise mathematical model of the process and on preenumerated symptom-fault patterns known as fault signatures. In the computer science/artificial intelligence community, model-based diagnosis relies on models of structure and behavior. For example, given symptoms of misbehavior (as detected by the behavioral model), fault candidates...
are identified using the structural model by following a dependency chain back from a violated prediction to each component and parameter that contributed to that prediction.

The model-based approach we discuss here has evolved within the AI community, but similar ideas have appeared independently in the fault detection and isolation literature. The type of model used in any model-based approach determines many of the capabilities and limitations of the specific method. Model types vary a great deal: There are numerical models, dynamic qualitative models, extended signed directed graphs, causal models and confluence equations, fuzzy qualitative models, and this article's semiquantitative model.

We focus here on process monitoring and diagnosis — the basic elements of an operator advisory system. In this setting, several conditions hold that challenge diagnostic methods. First, the plant is a continuous-variable dynamic system with feedback loops and changing states. Second, diagnosis must be performed while the system operates. Third, many system quantities are not sensed. Finally, measurements are unreliable due to sensor failures.

The basic idea: mimicry

The key cognitive skill for process operators is the formation of a mental model that not only accounts for current observations but also lets the operators predict near-term behavior as well as the effect of possible control actions. This observation underlies our framework for process monitoring, named Mimic. The basic idea is quite simple: Mimic the physical system with a predictive model, and when the system changes behavior due to a fault or repair, change the model accordingly so that it continues to give accurate predictions of expected behavior.

Figure 1 depicts the Mimic framework. Two tasks maintain the model. The tracking task advances the model’s state in step with observations from the physical system. When observations disagree with predictions, Mimic uses model-based diagnosis to determine the possible faults. After identifying a fault, the diagnosis task inverts it into the current model so that the predictions will continue to agree with actual observations. To be precise, Mimic maintains a set of candidate models since a given behavior might be caused by one of several faults. Each candidate model represents a possible condition of the system, including its state and faults.

The key benefit of this approach is that we can use the model as our window into the physical system. Specifically, the model can:

- detect early deviations from expected behavior more quickly than with fixed-threshold alarms (this method, known as analytical redundancy, uses known analytical relationships among sets of signals to check for mutual consistency);
- predict the values of unobserved variables (signal reconstruction) to permit alarms or other inferences on unseen variables, and to help the operator understand process conditions;
- predict near-term undesirable or hazardous conditions, thus providing early warnings; and
- predict the effect of proposed control actions to test if they will have the desired effect — a valuable capability in complex systems.

The end purpose of monitoring and diagnosis is advice to the operator about what’s happening and what to do about it. The advising task applies expert knowledge of safety conditions, recommended operating procedures, and performance objectives to produce advice in the form of alarms, warnings, and recommended actions. Although we do not discuss it further here, the advising task is a major beneficiary of the model-based approach in that the candidate models (and their tracked states) provide a testbed for generating warnings and for testing proposed control actions.

Three key technologies

Mimic exploits three relatively new technologies: semiquantitative simulation, measurement interpretation (tracking), and model-based diagnosis. These technologies work together in a hypothesize-build-simulate-match cycle, as shown in Figure 2. This figure gives a more detailed view of the tracking and diagnosis tasks of Figure 1.

Semiquantitative simulation. Industrial process plants (such as chemical refineries and nuclear power plants) are examples of continuous-variable dynamic systems. In modern control theory, these dynamic systems are modeled with a set of coupled first-order differential equations consisting of balance equations, physical-chemical state equations, and phenomenological laws. Given a set of initial values, a numerical simulation of the equations yields precise predictions of values for each variable over time.

Oddly enough, standard numeric simulation is too exact and too narrow for our purposes. Real process systems contain much imprecision. Sensors, actuators, and functional units operate within certain tolerances; parameter values and some functional relations are known only approximately. One approach, of course, is to do precise numerical simulation using average values, and then use some form of approximate matching of simulation results to observations. This approach presents two problems. First, given initial conditions, numerical simulation predicts only one behavior from a model, even though more than one might be possible, given the real imprecision. For example, a tiny difference in one parameter can determine whether or not a rocket achieves escape velocity. In effect, numerical simulation makes an inappropriate commitment to a single behavior. In contrast, qualitative simulation guarantees that all possible behaviors will be predicted. This capability is especially important in testing a fault hypothesis, which can exhibit several qualitatively distinct behaviors.

The second problem is the approximate-matching problem — how do you decide, in a principled way, when a difference between prediction and observation is due to imprecision and when it is due to a fault? What we really want is to explicitly express imprecise knowledge as part of the
model and have the simulator use it, producing valid ranges for each variable and permitting direct matching of observations to predictions. Semiquantitative simulation provides this capability. Furthermore, when observations match the predictions of two (or more) distinct behaviors, we want to track both hypotheses until they diverge, which Mimic does.

Qualitative simulation, the foundation for semiquantitative simulation, has two important characteristics for our application. First, it uses a qualitative level of description that lets us express imprecise knowledge. This purely qualitative description uses no numbers (but can take advantage of quantitative information when available). Second, qualitative simulation generates all the qualitatively distinct behaviors that are attainable from a starting state and consistent with the given imprecise knowledge. This property is essential in monitoring a physical system, whether healthy or faulty.

Semiquantitative simulation takes advantage of quantitative knowledge when it is available, which is always the case in process plants. This knowledge consists of numeric ranges for landmark values (for example, the pressure relief valve opens at 200-210 pounds per square inch) plus envelope functions that define the limits of monotonic relationships (for instance, an approximate relationship between the volume of fluid in a tank and its height). Qualitative values are expressed as numeric ranges and simulated with modified interval arithmetic. Interval arithmetic is normally subject to an uncertainty explosion, but this problem is avoided because all reasoning takes place with respect to the fixed set of landmarks provided by the qualitative behavior. The resulting semiquantitative simulation retains all the properties of qualitative simulation, but with two additional benefits: (1) it eliminates behaviors that are qualitatively possible but quantitatively inconsistent, and (2) it permits direct comparison of numeric sensor readings to the numeric ranges predicted for each variable. Qsim, which we have used in our research, provides this form of semiquantitative simulation.

The version of Mimic we describe here has evolved from an earlier design that based hypothesis generation on a decision tree and used a less sophisticated form of semiquantitative simulation.

**Tracking.** Mimic seeks to maintain a model whose state and faults (if any) reflect the current state of the physical system. More precisely, it maintains a set of models, each in a state consistent with the most recent observations. This set is called the "tracking set," and each model in the set embodies different fault hypotheses and therefore represents an alternate interpretation of the system. During diagnosis, Mimic adds models to the tracking set as it generates fault/repair hypotheses. During tracking, Mimic deletes models from the tracking set when their predictions fail to track observations.

Qualitative simulation generates a behavior graph, which is a directed graph of the system's possible qualitative states and the transitions among them. A behavior is a path through the directed graph and comprises a sequence of states alternating between states representing an instant of time and states representing an interval of time. Tracking, also called measurement interpretation, is the process of using observations to follow a path (a behavior) through the behavior graph.

Using the behavior graph fragment in Figure 3, we can describe several details of the process. If a model is in state E, Mimic compares each new set of observations to the values of state E. If the observations match the predictions, the model remains in state E. The usual noise filtering of sensor readings should still be performed before matching.

When an observation does not match the current state E, Mimic uses incremental simulation to generate the immediate successor states F, G, H. (Recall that a state in a semiquantitative simulation can have more than one possible successor state because of the imprecise knowledge expressed in the model.) If, say, the observations match state G, then the model is retained with its state now set to G. Incremental simulation refers to the control that the tracking task exerts over the simulator. When triggered, the simulator generates only the immediate qualitative successor states to the current state. Thus, the simulation is advanced only as needed; it is never "run to completion."

If the observations do not match any of the immediate successor states, Mimic repeats the incremental simulation and compares the observations to the second-generation successor states (I, J, K, L, M). The model needs this limited-distance look-ahead to jump over instantaneous states that fall between consecutive observations.
Observations can include independent variables, that is, exogenous variables whose values cannot be predicted. When an independent variable changes value, tracking recent predictions for integrated quantities. Thus the simulation, just like the physical system, is made to react to changes in independent variables.

Through progressive step-size refinements of the semiquantitative simulation, we can attain a desired precision for the quantitative predictions. Specifically, we can use the time interval between qualitative time points to refine the step size of the quantitative simulation, inserting new quantitative states that have time points within that interval. These new quantitative states more precisely bound the predicted behavior.

Mimic never has to generate the full behavior graph (envisioment), a computation that can be prohibitively expensive for complex systems because of the intractable branching problem of qualitative simulation. Instead, Mimic performs incremental simulation to generate only the states in the immediate vicinity of the last tracked state, abandoning any branches that do not track the observations. In effect, the observations act as a filter that eliminates irrelevant branches in the behavior graph.

An example

To illustrate Mimic at work, consider the electric water heater shown in Figure 4, which we have modeled and tested with Mimic. The water heater has a single thermostat that controls whether or not power is applied to the two heating elements (on/off control). Raw sensor information comes from a temperature sensor near the thermostat, a flow-rate sensor on the cold-water inlet, and from a voltage sensor on the heating elements. In a real monitoring situation we would want to diagnose various possible faults such as defective heating elements, a stuck thermostat, a faulty flow-rate sensor, and loss of electrical power. However, to keep this example simple, we consider only the possibility of defective heating elements.

The water heater is modeled as a two-compartment model in which two masses of water (upper and lower) are connected with thermal flow and mass flow between them, as shown in Figure 5. Each compartment is modeled as a two-compartment model: In the normal (fault-free) model all sensor measurements are consistent with the model variations. However, to keep this example simple, we consider only the possibility of defective heating elements.

Model-based reasoning avoids combinatoric problems in handling failed sensors and unavailable data because it matches against predictions rather than symptoms. When a sensor is bad and thus gives readings different from predicted, the sensor becomes a suspect because it is upstream of the discrepancy. If a datum is unavailable, it is not compared with predictions and therefore cannot cause discrepancies.
components (tank, heating elements, thermostat, temperature sensor, flow-rate sensor, voltage sensor, voltage supply) operate according to their intended purposes. In a fault model, a faulty component operates according to a failure mode, such as a heating element that generates no heat when power is applied.

Table 1 summarizes an example of monitoring the water heater, showing how monitoring progresses over eight moments in a series of observations. The observations are simulated from a separate numerical model of a faulty water heater in which the lower heating element produces no heat. For each moment, the table shows what hypotheses have been proposed and what models are being tracked. The water heater begins in a state where the tank’s water is hot, the heating elements are off, no water is flowing, and the temperature is slowly falling. These readings are consistent with the normal model. Now, someone starts to draw water for a bath. A high flow rate is measured, but all other readings remain the same. Since water flow is an independent variable, Mimic reinitializes every tracked model (just the normal model in this case) to reflect the change. Since the normal model remains consistent with the new values, it is retained.

As time continues, the temperature inside the tank drops because of the cooler inlet water. These readings are consistent with the current state of the normal model, so no change occurs to the tracking set. At moment 3 the temperature drops to the point where the heating elements turn on, as observed by the voltage sensor. Since this event is predicted by the normal model as an immediate successor state of the most recently tracked state, the normal model is retained with its state updated.

At moment 4 the temperature continues to drop. Although this observation is qualitatively consistent with the normal model, it is inconsistent with the associated quantitative ranges. In effect, the model is saying that for this flow rate, tank capacity, heating rate, and inlet temperature, the water temperature should not be dropping so fast. Thus, the tracking task discards the normal model. At the same time, this discrepancy triggers dependency tracing, which identifies two possible faults — a bad upper heating element or a bad lower heating element (denoted bad-h1 and bad-h2). This causes Mimic to build two fault models. Both models are successfully initialized, so Mimic is now tracking two models. Since Mimic assumes that only one fault occurs at a time, it does not hypothesize a double fault (bad-h1 and bad-h2). In a more detailed example, other hypotheses would also be proposed, such as a faulty temperature sensor, faulty flow sensor, and faulty thermostat, since all are upstream of the temperature discrepancy.

The water flow stops at moment 5 (somebody turned off the faucet). With this change in an independent variable, Mimic reinitializes the two models. At moment 6, the temperature is observed to be rising. Mimic then compares the observed temperature to the quantitative predictions of the two models. Because the observed temperature exceeds the range predicted by the bad-h1 model, Mimic discards that model. The predictions of the one remaining model, bad-h2, are compatible with the observations, so the model is retained. This model continues to track future readings, and emerges as the sole fault hypothesis.

<table>
<thead>
<tr>
<th>MOMENT</th>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>TIME (MIN.)</strong></td>
<td>0.0</td>
<td>1.0</td>
<td>2.0</td>
<td>2.4</td>
<td>2.7</td>
<td>3.0</td>
<td>13.0</td>
<td>27.7</td>
</tr>
<tr>
<td><strong>FLOW (LITERS/Min.)</strong></td>
<td>0</td>
<td>30</td>
<td>30</td>
<td>30</td>
<td>30</td>
<td>30</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td><strong>TEMPERATURE (deg. C)</strong></td>
<td>64.9</td>
<td>64.6</td>
<td>61.4</td>
<td>58.9</td>
<td>57.1</td>
<td>55.9</td>
<td>60.0</td>
<td>66.0</td>
</tr>
<tr>
<td><strong>POWER (ON OR OFF)</strong></td>
<td>off</td>
<td>off</td>
<td>off</td>
<td>on</td>
<td>on</td>
<td>on</td>
<td>off</td>
<td></td>
</tr>
<tr>
<td><strong>NEW FAULT HYPOTHESES</strong></td>
<td>none</td>
<td>none</td>
<td>none</td>
<td>none</td>
<td>bad-h1</td>
<td>bad-h2</td>
<td>none</td>
<td>none</td>
</tr>
<tr>
<td><strong>NEW FAULT MODELS</strong></td>
<td>none</td>
<td>none</td>
<td>none</td>
<td>bad-h1</td>
<td>bad-h2</td>
<td>none</td>
<td>none</td>
<td>none</td>
</tr>
<tr>
<td><strong>TRACKED MODELS</strong></td>
<td>normal</td>
<td>normal</td>
<td>normal</td>
<td>bad-h1</td>
<td>bad-h2</td>
<td>bad-h1</td>
<td>bad-h2</td>
<td>bad-h2</td>
</tr>
</tbody>
</table>

Figure 5. A structural model of water heater.

Table 1. Diagnosing the water heater based on its dynamic behavior.
Advantages

The water heater example shows how Mimic can diagnose a system by observing its dynamic behavior with few observable variables. In general, the speed at which Mimic can narrow its diagnosis depends on the number of monitored variables and the dynamic activity of the system. With more monitored variables and more system activity, more opportunities arise to refute incorrect hypotheses.

Diagnostic systems often rank competing hypotheses by probability, based on the previously known fault probabilities of components. Because Mimic monitors continuously, it can also rank hypotheses by age. The longer a hypothesis survives changing observations, the stronger the evidence supporting that hypothesis. This age-ranking also focuses attention on hypotheses that account for the earliest manifestations of a fault, before other manifestations and corresponding hypotheses appear. In short, the system’s natural time delays help identify the correct hypothesis.

Mimic treats alarms in a new way, in that they are based primarily on the predictions of the models in the tracking set. This has several positive consequences:

- Alarm thresholds can be dynamic rather than fixed, thus allowing earlier alerting.
- Alarms can be based on unobserved variables, permitting more freedom in alarm design.
- Alarms can reveal any mutually inconsistent readings (extreme analytical redundancy).
- Alarms, called forewarnings, can be based on near-future predicted states.
- False alarms due to operating-mode changes (for example, startup or shutdown) should not occur if the model faithfully predicts such dynamic behavior.

Limitations

If Mimic cannot quickly refute invalid hypotheses, the tracking set will grow and Mimic will slow down correspondingly. Mimic refutes hypotheses through tracking: there must be an observed discrepancy between the model’s predictions and the sensor readings. In practice, this means the model’s quantitative predictions must be reasonably well bounded and that there must be an adequate number of well-placed sensors.

Mimic assumes that faults occur one at a time. More precisely, it assumes that the manifestations of different faults appear at different times with respect to its sampling rate. This assumption can be violated in the case of a catastrophic event (such as an explosion) or cascading faults, where discrepancies due to more than one fault can appear simultaneously.

A qualitative simulation algorithm can predict spurious behavior when the qualitative model does not explicitly preserve an invariant quantity (for example, energy) from a qualitative state to its successors. If the manifestations of an actual fault happen to match a predicted but spurious behavior, then the system will find no discrepancy and the fault will go undetected. However, this problem has been substantially reduced by introducing several global constraints, that is, constraints that eliminate spurious behaviors through global consistency checks, such as applying the nonintersection constraint to trajectories in qualitative phase space, automatically deriving energy constraints by recognizing conservative and nonconservative forces, and using higher-order derivative constraints.

The Qsim algorithm guarantees that all behaviors are predicted, and only under a qualitative level of description does this give a tractable set of possibilities. In simple cases such as the water heater, this is tractable in practice as well as in theory. For more complex systems, controlling the size of the hypothesis set is still a potential problem.

Related work

Kay has demonstrated the Mimic approach in monitoring the pump-down phase of a vacuum system for semiconductor fabrication, which requires ultrahigh vacuums ($10^{-9}$ Torr). Since no practical theory exists for the sorption of gases, it is difficult to model the process numerically. Kay’s semiquantitative model, with dynamic envelopes that bound the expected observations, permits reasoning with uncertainties and still achieves detection of faults early in the pump-down phase.

Abbott’s approach to monitoring and diagnosis, like Mimic, takes advantage of the sequence in which symptoms appear, although the mechanisms differ somewhat. Her Draphys system detects symptoms (discrepancies) by comparing sensor readings to expected values computed from a numerical simulation model of the fault-free system. Fault hypotheses are then generated by tracing upstream from the symptom through a graph model of the paths of interaction among components, tracing both functional and physical paths. As new symptoms appear, Draphys tests each existing hypothesis to see if propagating its effects further downstream in the graph model covers the new symptoms. This latter step is akin to Mimic’s tracking, but at a more abstract level. Specifically, the graph model in Draphys represents only that a fault in one component might affect another component; there is no information about whether the affected sensor should read high or low, nor about time delays in fault propagation. Such information could be used to refine some hypotheses. Mimic can refine some hypotheses because the semiquantitative model it uses during tracking provides such information.

Isermann’s model-based approach to process fault diagnosis, like Mimic, uses dynamic mathematical models and measurable input and output signals to allow estimation of unmeasurable internal quantities, which can then be used for fault detection. Unlike Mimic, however, Isermann’s models are strictly quantitative and are expected to “describe the process behavior precisely.” The resulting approximate-matching problem (to determine if an observation is “normal”) is handled with a Bayes decision algorithm. After recognizing a symptom, this system classifies the fault by comparing it with fault signatures, which have been established beforehand. Although Isermann’s work uses different methods for simulation, measurement interpretation, and diagnosis, he reaches a
conclusion that we share: "Dynamic process behavior yields considerably more information on process faults than can be achieved in the static case."

Several expert systems have been built that share Mimic's operational goal — that of relieving some of the monitoring burden from process operators. Mimic focuses primarily on determining the physical system's state, but most of these expert systems have the broader scope of trying to advise the operator on corrective actions. ESCORT, for example, gets its knowledge of faults, anomalies, and corrective actions through the usual process of codifying human expertise in rules, rather than by encoding a predictive model of the physical system as Mimic does.

**COMPARSED TO EXISTING METHODS**

Based on fixed-threshold alarms, fault dictionaries, decision trees, and expert systems, our method for monitoring and diagnosing process systems has several advantages beyond those we've already discussed:

- A semiquantitative model of a physical system can predict all possible behaviors that are consistent with the incomplete and imprecise knowledge of the system's devices and processes. This ensures, for example, that rare but hazardous behaviors will not be overlooked.

- By using a structural model of the plant and tracing upstream from the site of unmatched observations, model-based diagnosis generates fault candidates efficiently, without resorting to precompiled (and often incomplete) symptom-fault patterns.

- By injecting a hypothesized fault into the model and tracking its predictions against observations, the dynamic behavior of the plant is exploited to corroborate or refute hypotheses.

- By simulating ahead in time from the current state, an operator can be warned of partial quantitative information and permit automatic injection of faults.

**Acknowledgments**

We thank the guest editors and anonymous referees for pointing out several shortcomings in the earlier version of this article. Daniel Dvorak has been supported by the AT&T Doctoral Support Program. Benjamin Kuipers has been supported in part by NSF grants IRI-8602665, IRI-8905494, and IRI-8904454, by NASA grants NAG 2-507 and NAG 9-200, and by the Texas Advanced Research Program under grant 00365817.

**References**


**Further reading**

D. Dalle Molle, Qualitative Simulation of Dynamic Chemical Processes, doctoral dissertation, Univ. of Texas at Austin, 1989.


Q. Shen and R. Leitch, "Synchronized Qualitative Simulation in Diagnosis," to be published in Working Papers Fifth Int'l Workshop Qualitative Reasoning about Physical Systems, AI Lab, Univ. of Texas at Austin, 1991.


Daniel Dvorak is a distinguished member of technical staff at AT&T Bell Laboratories and a doctoral candidate in computer science at the University of Texas at Austin. His research interests include model-based reasoning, qualitative reasoning, and case-based reasoning. Recent work has focused on knowledge-based monitoring of the UUCP computer network.

Dvorak received his BS in electrical engineering from Rose-Hulman Institute of Technology in 1972 and his MS in computer engineering from Stanford University in 1974. He is a member of the IEEE Computer Society, IEEE, AAAI, ACM, and Computer Professionals for Social Responsibility.

His address is AT&T Bell Laboratories, 2000 N. Naperville Rd., Naperville, IL 60566-7033; e-mail, dvorak@ihlpz.att.com

Benjamin Kuipers is an associate professor of computer science at the University of Texas at Austin in 1970 and a member of IEEE Expert's editorial board (this photo appears on p. 2). His research interests include commonsense knowledge, qualitative reasoning with incomplete knowledge, resource-limited inference, and spatial exploration, learning, and problem solving.

Kuipers received his BA in mathematics from Swarthmore College in 1970 and his PhD in mathematics from the Massachusetts Institute of Technology in 1977. He is a member of AAAI, ACM, the Cognitive Science Society, and the New York Academy of Sciences.

His address is Department of Computer Science, University of Texas at Austin, Austin, TX 78712; e-mail, kuipers@cs.utexas.edu