BladeCenter thermal diagnostics

W. J. Piazza
R. E. Harper
M. J. Crippen
J. A. Matteson

An analytical technique called thermal diagnostics is presented as a tool for determining the root cause of thermal anomalies arising in electronic equipment. The technique utilizes a dynamically constructed flow network model, real-time inventory, temperature, utilization metrics, and statistical hypothesis testing to select the most likely scenario from among thousands of potential causes of thermal problems. This paper describes the concept of thermal diagnostics and concludes with results from a laboratory evaluation in which we physically trigger thermal anomalies on a running IBM eServer™ BladeCenter® system and record the diagnosis given by the algorithm. In these tests, our algorithm correctly diagnosed the thermal situation and provided meaningful guidance toward clearing the detected problems.

Introduction

The IBM eServer* BladeCenter* system provides a superb platform for product development and research to collaborate on projects that increase the ability of a server to autonomically respond to changing conditions. For example, IBM researchers and engineers have demonstrated that with the proper software in place, it may shift workload from one server to another to manage overall power consumption in a data center [1] or to help deal with the problems associated with software aging [2]. Companion papers in this journal provide an overview of the system [3] and describe the systems management software [4] and packaging, power, and cooling infrastructure [5]. The remainder of this paper describes work done jointly by IBM Research and IBM xSeries* Product Development pertaining to the ability to detect the early onset of thermal problems that affect electronic equipment and to determine their root causes in a timely and cost-effective manner.

For several reasons, the BladeCenter system is particularly well suited for the type of thermal analysis described here. In the BladeCenter environment servers share hardware resources, including enclosures, power supplies, blowers, and management hardware—all of which contribute in some way to the thermal conditions to which each of the otherwise independent servers is exposed. Also, the system promotes such a high spatial density of servers that the management of power consumption and cooling becomes an essential part of systems management. In addition, because of the large number of subsystems housed within a BladeCenter enclosure, the airflow and heating patterns are more complicated than those of previous-generation servers.

A thermal crisis is defined here to be a situation in which a monitored temperature within a piece of electronic equipment is reportedly approaching a critical threshold beyond which the equipment should not be operated. In electronics equipment in general, a thermal crisis may result from a number of different causes. Among them are the following:

- Overheating of an electrical component within the enclosure due to a failure of that component (e.g., a short circuit).
- Partial or total failure of a cooling fan within the enclosure.
- Failure of external heating, ventilation, and air conditioning (HVAC) equipment.
- Inadequate cooling of the facility (e.g., an overcrowded data center).
- Airflow blocked by an outside obstruction (e.g., close proximity to a wall, or materials placed in front of air intakes).
- Removal (without replacement) of an enclosure panel or a pluggable subsystem (causing a redistribution of airflow within the enclosure).

0018-8646/05/$5.00 © 2005 IBM

- Over-configuration (i.e., violation of enclosure specifications regarding limitations for some configurations).
- Erroneous temperatures reported by thermal sensors within a subsystem.
- Buildup of contamination on air filters.

While the BladeCenter system was designed to eliminate or reduce the likelihood of these types of problems, it is still possible that a malfunction or operator intervention may circumvent a defensive mechanism. For example, it is designed with pop-up dampers to prevent diverted airflow from causing problems in peer subsystems when a processor blade is removed for replacement. However, a broken damper or the action of an operator manually preventing the motion of the dampers could create an undesirable operating environment. Anecdotal input from customers suggests that painters' drop cloths hung in front of racks, boxes piled in front of equipment, defective sensors, and air conditioning failures at inopportune times are more common than any of us would like to believe; thermal diagnostics can detect and initiate a response to each of these events.

To prevent heat-related damage to equipment, it is a common practice in the industry to provide internal temperature monitoring and to implement a dual-level threshold scheme. When a temperature reading is seen to cross a first threshold (e.g., 85°C), a warning message is generated to notify the operator. Then, if any sensor reading crosses a second (error) threshold (e.g., 95°C), the equipment will shut down. In the present state of the art, a major component near the sensor producing an unacceptable reading is assumed to be responsible for the situation, and minimal effort is made to isolate the problem to one of the root causes listed above. A more detailed analysis and root-cause determination could result in faster reconciliation of the problem, lower warranty costs, and a broader range of autonomic responses to the problem. The type of analysis described here allows problems to be detected sooner (resulting in more time to respond) and may allow management software to estimate how long it will take before the situation reaches a critical point. Some examples follow:

- If it is known at the time the warning is sent that a cover panel has been removed from the enclosure (leaving an opening that alters the airflow within the enclosure), workload could be shifted to servers known to be outside the influence of that side panel, and a hardware technician could be dispatched and given explicit instruction on how to correct the problem.
- If it is known that the problem is caused by blocked airflow to a set of processor blades in a BladeCenter environment, software could automatically reassign workload to unaffected processor blades and reduce the amount of workload assigned to the obstructed blades.
- If the source is localized to a particular subsystem that is overheating, the problematic subsystem can be throttled or varied offline, and a request to service the subsystem can be initiated.
- If it is known that the reported problem is due to a malfunctioning temperature sensor, a decision can be made to continue normal operation, as opposed to shutting subsystems down and reducing the system capacity or throughput.
- Certain types of problems may be alleviated by increasing the speed of fans or blowers, whereas for other types of problems it may be clear that such intervention is pointless and would only increase acoustic noise levels. Knowing the specific root cause can assist in determining when it is beneficial to change fan speed.

This paper describes a method of performing thermal diagnostics on a complicated piece of electronic equipment with the goal of isolating a thermal crisis to a specific root cause and taking an appropriate action.

Technical approach

The present work builds upon a modeling technique known as *flow network modeling* (FNM), a tool currently used by design engineers as they study airflow and heat buildup within an equipment enclosure during the equipment design phase.

FNM techniques are traditionally applied to hardware that is being developed, not to in-service equipment. FNM techniques can be used to examine a static configuration of the hardware with the goal of determining whether design changes—such as a larger fan, larger ventilation slots, internal baffles, and so on—are required in order to cool the equipment properly in some supported environment. Once the system design is such that FNM indicates that there are no thermal issues for all desired configurations of properly functioning hardware, the thermal design is considered complete. In contrast, this paper discusses a new dynamic use for FNM as it is applied to production equipment installed in the field and in use.

FNM background

FNM allows temperatures throughout a system to be predicted on the basis of knowledge of airflow patterns and power dissipation within the system enclosure. First,

the airflow impedances of individual subsystems and plenums within the enclosure are determined. These impedances can then be combined in a prescribed manner to produce an estimate of the total system impedance to airflow. Next, the rate of airflow through the enclosure is determined. This is done by finding the intersection of the characteristic curve of the cooling fans and a line representing the pressure drop through the entire system, which is based directly on the system impedance. Once this system airflow rate and a corresponding pressure drop are known, they are divided among the individual subsystems so that the airflow through individual subsystems becomes known. Then, the airflow through a subsystem and the power dissipated by the subsystem can be used to determine the amount of temperature rise that will occur in that subsystem. The operational temperature of any subsystem can therefore be estimated by following the flow of room-temperature air through the system and accumulating temperature rises each time the air passes through a subsystem. For a more complete description and examples of basic flow network modeling, see [6, 7].

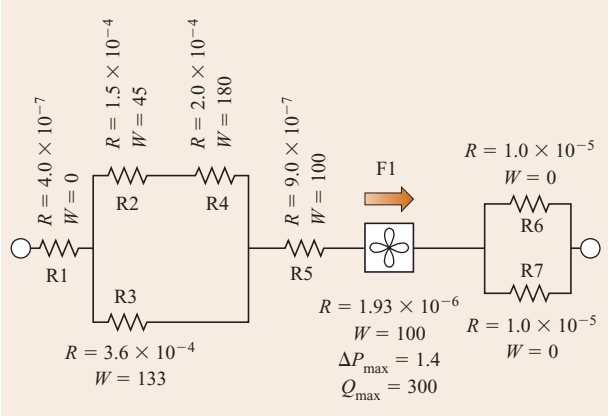

Figure 1

Schematic airflow diagram (R values are given in units of "in $H_2O/(ft^3/min)^2$," W in watts, ΔP_{max} in units of "in H_2O," and Q_{max} in units of ft^3/min).

Overview of thermal diagnostics

We define the term *thermal diagnostics* to include the steps of collecting vital information about a system (including inventory and performance and temperature metrics), constructing and analyzing a flow network model of the specific system configuration (including thousands of problem scenarios injected into the model), determining a most-likely scenario, and initiating an appropriate response based upon the diagnosis available after analysis.

A thermal diagnostic scan can be initiated periodically or in response to the detection of a pending thermal crisis (e.g., by observing that a temperature threshold value has been exceeded or that a temperature reading is trending upward over time). During a thermal diagnostic scan, software uses an identifying number (such as the part number or machine type or model number) of the chassis and each installed subsystem to look up a *thermal model* describing that component. These thermal models are currently implemented as XML files stored on the fixed disk of some management server. The thermal diagnostic software then constructs a flow network model of the entire chassis using the information contained in the individual thermal model files, and it is this chassis-level flow network model that becomes the basis for the rest of the analysis.

Once the chassis-level flow network model exists, it may easily be manipulated to simulate nominal and problematic scenarios. For example, the flow resistance of one or more air pathways can be increased to model an obstruction, or the wattage dissipated by a component can be increased to simulate a component failure. We currently model nominal conditions, intake obstructions, overheating components, defective temperature sensors, impaired fans, and missing chassis panels. For each problem scenario, the problem may be applied to a number of different locations within the chassis and to varying degrees of severity (e.g., an obstruction covering only 25% of an intake vent, 30% of the intake, 35%, and so on). In addition, each scenario is run a number of times while the room ambient temperature is assumed to vary by small steps over the operating range of the chassis. In the end, the model may test 100,000 or more different combinations of these variables. Because of the large number of permutations that could result, we limit the model to testing a maximum of one problem scenario applied to the chassis at a time. The algorithms and processes described herein may be codified as software in a number of forms, including a standalone diagnostic tool or incorporated into some management infrastructure, such as IBM Director, a remote service adapter, BladeCenter management module, or any software provided to monitor the health of the system. Our current prototype is implemented as an extension of IBM Director and uses Director to initiate the diagnosis, collect information about the target chassis, and respond appropriately to the diagnosis.

Details of the FNM methodology

This section describes in detail the steps used to construct and analyze the flow network model that is the heart of thermal diagnostics. We illustrate our approach using a simple example outlined in **Figure 1** and further elaborated in subsequent figures. Figure 1 schematically

shows the arrangement of a number of air paths and air-moving devices in a fictitious piece of equipment.

The first step in constructing the model is to enumerate all hardware elements through which air may flow (such as grillwork, plenums, air channels, spaces between circuit boards and other elements) and air-moving devices, such as fans and blowers. For each hardware element, we must calculate the impedance (R) to airflow. The available literature on FNM techniques (e.g., [6]) includes a number of formulas that allow designers to estimate airflow impedance. The formulas take into account physical dimensions, grillwork and slots, turns, smoothness of surfaces, and so on. Impedance to airflow is reported in units of "in $H_2O/(ft^3/min)^2$" or its metric equivalent. [This unit represents the rise (in inches) of a column of water due to the change in static pressure across the impeding element for a specific volume of airflow.] As an alternative to being estimated, the impedance may be predicted using computational fluid dynamics modeling software, or it may be measured on actual hardware.

For air-moving devices such as fans and blowers, we must identify the maximum airflow (Q_{max}) that the device can produce and the maximum static pressure drop (ΔP_{max}) across the device. (These values are typically obtained from the manufacturer's specification sheet.)

For each element that produces heat, we must identify the maximum power (W_{max}) and, in some cases, the minimum power (W_{min}) dissipated by the element. W_{min} is important if the wattage dissipated by the element varies over a wide range during normal usage. The physical relationships of serial and parallel air paths shown in Figure 1 can also be illustrated using a tree structure, as shown in **Figure 2(a)**. This tree structure is, in fact, used by the thermal diagnostics software because it is a convenient data structure for representing the airflow model and for controlling the order of subsequent calculations that must be performed using the model. Leaves (endpoint nodes) in this tree represent the physical elements that move or impede airflow, such as plenums, fans, and circuitry. Parent nodes of various types allow leaf nodes to be combined in parallel or serial fashion. It is important for some of the steps that follow that serial (S) nodes keep track of the ordering of the elements under them.

To build the tree properly, the software must be able to detect serial and parallel configurations of elements. Serial configurations are detected by looking for nodes in the network that are shared by exactly two nonterminal elements. Parallel configurations are detected by looking for elements that share the same nodes at both ends.

We compute R_{sys} (the impedance of the entire system) by applying the following rules to nodes throughout this tree structure:

a. At serial nodes, the impedances of all child nodes are simply added (see [7]):

$$R = R_1 + R_2 + R_3 + \cdots , \tag{1}$$

where R is the total impedance of the series of elements and R_1, R_2, etc. are the impedances of individual child elements.

b. At parallel nodes, the impedances of child nodes combine according to this formula (see [7]):

$$\frac{1}{\sqrt{R}} = \frac{1}{\sqrt{R_1}} + \frac{1}{\sqrt{R_2}} + \frac{1}{\sqrt{R_3}} + \cdots . \tag{2}$$

Equation (2) is not the more familiar formula for parallel resistors used in electronic circuits because, unlike the linear relationship among voltage, current, and electrical resistance, airflow impedance is related to pressure drop and to airflow volume via the formula shown in Equation (3) (see [7]):

$$\Delta P = R \times Q^N . \tag{3}$$

The exponent N in Equation (3) varies from 1.0 for purely laminar flow to 2.0 for purely turbulent flow. Our work currently assumes completely turbulent flow, which appears to be a reasonable assumption for high flow rates in a complicated chassis with electronic components in the pathways. The current model gives good results using $N = 2$, and future work could investigate whether refinement of this value is warranted.

The values for underlying nodes of an S (serial) node or a P (parallel) node in the tree can be combined using these rules, and the computed value becomes the R value for the parent node. The computation of R values propagates up through the tree until the root node value is known; the R value of the root node is then designated as R_{sys}.

As we work through the tree structure to compute R_{sys}, we can also compute system-level values of Q_{max} and ΔP_{max} from all of the air-moving devices in the enclosure. The following rules apply:

- At S nodes: Add underlying ΔP_{max} values for all air-moving child nodes. This reflects the fact that when air-moving devices are in series, they produce an increased pressure drop.
- At P nodes: Add underlying Q_{max} values for all air-moving child nodes. This reflects the fact that when air-moving devices are in parallel, they produce an increased airflow volume.

Note that these rules are valid if serial or parallel configurations of air-moving devices use matching fans or blowers. This is a reasonable approximation for the BladeCenter design, in which two identical blowers drive

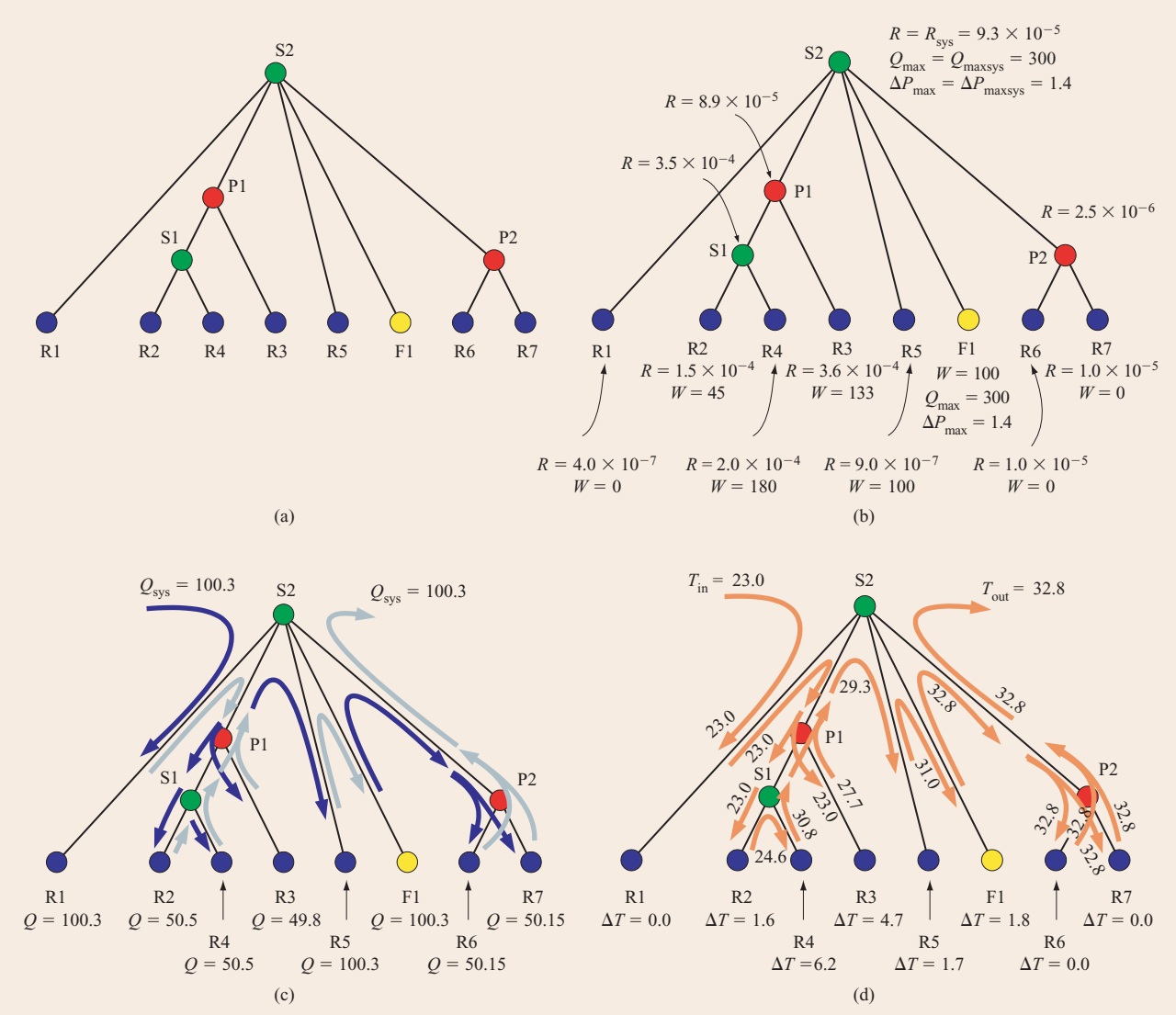

Figure 2

(a) Tree structure corresponding to airflow model. Blue nodes represent the airflow resistance of individual elements; yellow nodes represent air-moving devices; green nodes represent the aggregation of elements along a series path; red nodes represent the aggregation of elements along a parallel path. (b) Tree structure with values propagated up according to rules. (c) Distribution of airflow within the enclosure. At green (series) nodes, each underlying element sees the same air-stream volume and passes heat to the next underlying node; at red nodes the air stream splits and is heated to varying degrees along each parallel path before mixing together to reform a single air stream. (d) Temperatures throughout the enclosure are derived from some assumed ambient temperature by accumulating the contribution of each device in the air path.

all air movement; further refinement of the rules could allow for dissimilar air-moving devices.

Like impedance values, the computations of Q_{max} and ΔP_{max} propagate up through the tree structure, and the values computed for the root node are designated Q_{maxsys} and ΔP_{maxsys} [**Figure 2(b)**]. We must next compute the operating point of the system (Q_{sys}) by finding the intersection between the characteristic curve of the air-moving devices and the enclosure resistance curve. This is often a graphical step that involves plotting the enclosure curve, i.e., Equation (3), over the characteristic curve of the air-moving device. In the thermal diagnostic software, a calculation is used to find the approximate operating point of the system:

a. The characteristic curve of the combined air-moving devices is approximated as a straight line defined by Q_{maxsys} and ΔP_{maxsys},

$$Q = Q_{\text{maxsys}} \times \left[1 - \frac{\Delta P}{\Delta P_{\text{maxsys}}}\right] \tag{4}$$

or

$$\Delta P = \Delta P_{\text{maxsys}} \times \left[1 - \frac{Q}{Q_{\text{maxsys}}}\right], \tag{5}$$

where Q is the airflow volume for some corresponding pressure drop ΔP and Q_{maxsys} and ΔP_{maxsys} are the maximum airflow volume and maximum pressure drop that the air-moving devices can produce. Note that $Q = 0$ when $\Delta P = \Delta P_{\text{maxsys}}$ and $\Delta P = 0$ when $Q = Q_{\text{maxsys}}$.

b. The enclosure resistance curve is approximated by Equation (3) using $N = 2$ and R_{sys} as the value for R.

The overall system operates at the point at which the static pressure drop created by the air-moving devices and the static pressure drop through the enclosure are the same. Setting Equation (5) equal to Equation (3) and solving for Q (which is the operating point airflow denoted by Q_{sys}) gives

$$Q_{\text{sys}} = \frac{\frac{-\Delta P_{\text{maxsys}}}{Q_{\text{maxsys}}} + \sqrt{\left(\frac{\Delta P_{\text{maxsys}}}{Q_{\text{maxsys}}}\right)^2 + 4R_{\text{sys}}\Delta P_{\text{maxsys}}}}{2R_{\text{sys}}}. \tag{6}$$

By using the impedance values from the previous example, Q_{sys} is calculated to be 100.3 ft^3/min.

Q_{sys} defines the total airflow through the enclosure. This system-level airflow can be split up among child nodes to determine the airflow through each portion of the enclosure. To do this, our software traverses the tree [**Figure 2(c)**] and does the following:

- At S nodes, the Q value of the parent S node becomes the Q value for all child nodes, since all S nodes see the same airflow.
- At P nodes, the airflow is divided among all child nodes by noting that all P nodes experience the same pressure drop. The portion of air that a given child node will be allocated is derived from Equation (5) by setting ΔP to be the same for the combined parallel path (i.e, the parent node) and for any individual child path:

$$Q_{\text{child}} = Q_{\text{parent}} \times \sqrt{\frac{R_{\text{parent}}}{R_{\text{child}}}}. \tag{7}$$

Once the airflow (Q) is known for any element, the temperature rise (in °C) across that element can be estimated as (from [7])

$$\Delta T = \frac{1.76 \times W}{Q}, \tag{8}$$

where W is the power dissipated in watts and Q is the airflow volume in ft^3/min.

Equation (8) assumes that the system is at sea level; the factor of 1.76 can be adjusted upward if the altitude is known to be higher. For example, a factor of 2.0 roughly corresponds to a 7,000-ft altitude. This formula calculates the increase in temperature that can be expected in air passing over a power-dissipating component. Given an inlet temperature and the temperature increase over a component, it is possible to compute the local air temperature immediately downstream of a component.

Once the temperature rise across each component is known, we can determine actual temperatures throughout the entire chassis. To do this, we start with an assumed ambient temperature outside the chassis and follow the flow of air through each air pathway inside the chassis. The root node is given an *input temperature* that is the assumed room ambient temperature, and the *output temperature* that it computes is the exhaust temperature from the rear of the enclosure (assuming front-to-back airflow).

At S nodes, the input temperature to the S node is passed to the first child node, and the value of that child node output temperature is used as the input temperature for the next child node in the series. This repeats until all child nodes have contributed to the temperature rise. The output temperature reported by the last child node becomes the output temperature that the S node reports to its parent.

At P nodes, the input temperature to the P node is passed to each child node as an input temperature. The output temperature of the P node is computed as the weighted average of all child node output temperatures. The airflow volume (in ft^3/min) of each child node is used as the weighting factor. Note that this assumes total mixing of air. At leaf nodes, the output temperature is simply the input temperature plus the temperature rise within that element due to wattage dissipated by the element [**Figure 2(d)**].

In reality, the model performs calculations twice, once using the minimum predicted input temperature and the element W_{min} dissipation, then again using the maximum predicted input temperature and the element W_{max} dissipation. This results in two output temperature values corresponding to the expected range of temperatures in the subsystem, and both temperatures are propagated through the rest of the model.

Each time that a temperature sensor is encountered during the traversal of the tree structure, the *figure of error* metric associated with the scenario must be updated. The figure of error represents the similarity

between all expected temperatures for that scenario and all actual observed temperature readings. We compute the figure of error using a simple but effective *sum of squared errors* approach. Specifically, for each temperature reported to us by the hardware, we compute an error term (which is the distance that the reported temperature lies outside the expected temperature range predicted by the model), square the error terms, and sum them. The resulting metric has the desirable characteristics of being insensitive to the direction (high or low) of the error and climbing exponentially for larger error distances.

Finally, from among the many that are examined, we select one or more scenarios as the most likely thermal diagnosis. The selected scenario or scenarios are those for which the FNM-computed readings most closely match the observed values (as determined by the smallest figure of error).

Applying FNM as a diagnostic tool

With core software capable of analyzing hardware configurations using FNM techniques, a thermal diagnostics infrastructure can be built around it. The infrastructure includes the following elements:

1. The values of R (airflow impedance) and W (power dissipated) for all subsystems and the values of $\Delta P_{\max}$ (maximum static pressure) and $Q_{\max}$ (maximum airflow) for air-moving devices must be available to the diagnostic software. A number of approaches are possible. For example, the diagnostic software may provide its own data files or database of identification numbers for enclosures and pluggable subsystems. The R, W, $\Delta P_{\max}$, and $Q_{\max}$ values needed for the analysis can then be obtained from this source. Our current prototype uses a set of XML-formatted data files to convey this information to the diagnostic software. The R, W, $\Delta P_{\max}$, and $Q_{\max}$ values may be built into the enclosures and subsystems by making them available as entries in vital product data (VPD) erasable programmable read-only memory chips (EPROMS).
2. The topology (i.e., airflow paths) of the enclosure must be represented in the diagnostic software. Once again, it is possible to look this up in a data repository or build it into the hardware as data structures contained in nonvolatile memory. Our current prototype also extracts this information from XML files.
3. The diagnostic software must assess what manipulations of the R, W, and topology information make sense in the context of simulating potential thermal problems. For example, it may be much more likely that an obstruction would be placed at the intake vent of a processor blade than at some intermediate air path inside the blade. We provide hints to the model in the form of special XML tags that indicate where it makes sense to look for obstructions.
4. The diagnostic software must also assess the circumstances that will exist if a pluggable subsystem is not installed, including such cases as a missing filler panel. Again, we provide special XML tags to point to thermal models that should be applied for cases of both installed and missing filler panels.
5. Operational parameters such as temperature sensor readings, the speed of all air-moving devices, and processor utilization measurements must be collected and made available for the diagnostic software to use as needed. We have also found that it is important to know which operating system or hypervisor is running on the hardware because of differences in the methods used to measure the processor utilization under various operating system environments.
6. The diagnostic software must attempt to find a scenario that closely matches the observed temperature readings using the current hardware configuration as a starting point. Upon being notified that it is necessary to determine the root cause of a thermal crisis, the thermal diagnostics performs the following actions:
   a. Builds a software model (i.e., the tree structure described previously) that corresponds to the present hardware configuration. This is called the *base model*.
   b. For each of a number of predetermined scenarios, builds a new model (again, a tree structure) that corresponds to the base model plus the change introduced by the scenario. Scenarios can include the removal of one or more filler modules, the partial or total blockage of one or more intake vents, the overheating of one or more subsystems, partial or total fan failures, etc. Some scenarios may alter the values of parameters such as an impedance (R value) or power dissipated (W value), while other scenarios may require changes to the topology (such as the removal of a subsystem and replacement with an *open bay* component). Some scenarios may be rejected outright (e.g., a scenario for an IBM BladeCenter system that requires the overheating of processor blade 6 if it is known that processor blade 6 is not installed or not powered).
   c. For each valid scenario, the software must perform the FNM analysis of the resulting model

and compute a figure of error that indicates how well that model matches the observed readings.

d. The software selects the scenario that produces a figure of error that most closely matches the observed readings. Note that more than one scenario may result in the same figure of error, indicating that a single root cause cannot be determined.
e. Once a root cause is known, the diagnostic infrastructure must translate this information into an appropriate response. In our prototype, we use the IBM Director event-handling capabilities to invoke a response. The response may include administrator notifications in the form of a console message, page, or e-mail, or may be of a more autonomic nature, such as starting and stopping software running on affected blades and powering down compromised blades. The set of responses could easily be extended to include options such as increasing fan speeds, invoking subsystem power management features, and steering workload away from problem areas.

Dealing with variability

Thermal diagnostics do not run in a static environment where the ambient temperature is constant, where fan speeds never change, or where processor utilization is known in advance. Therefore, each of these variables must be factored into the analysis.

As previously stated, we assume that we do not know ambient room temperature (even though an ambient temperature sensor is provided on the chassis), and we model the chassis temperatures for a number of different assumed ambient temperatures. Then, when a best match is found, we infer not only the condition (such as an obstruction) that may exist, but also a most likely ambient temperature. Performing the analysis in this way allows us to catch defective ambient temperature sensors.

Fan speeds are reported to the analysis software, and the model is adjusted to provide the corresponding volume of airflow. As fan speeds change between attempts to diagnose the chassis, the model also adjusts. If the fan speed sensor is defective, the modeled volume of air will be wrong throughout the chassis, and nominal scenarios will provide a poor fit with actual temperatures. Scenarios that specifically model fans that are moving a different amount of air than they report provide a better match and therefore point to a fan problem.

Each subsystem may produce a variable amount of heat, depending upon the level of system activity. For example, in a BladeCenter system an individual processor blade may dissipate hundreds of watts when it is extremely busy and less than half that amount when it is idle. This has a direct effect upon the temperatures within that subsystem, and the model must take this into account to make accurate predictions.

Processor utilization metrics can be used to refine the estimate of the amount of heat being dissipated by some components. In fact, utilization metrics are so important that thermal diagnostics may choose to discard some problem scenarios if utilization data for selected subsystems is not available.

Most sources of processor utilization data are dependent upon the operating system or hypervisor in some way. We have learned that it is important to understand exactly how and when utilization data (and corresponding temperatures) are collected. For example, the reported values may be instantaneous readings, or they may be a rolling average over some period of time. If averaged, data points may have been weighted in some special way. We have also seen operating systems and hypervisor implementations that may themselves drive the level of processor activity quite high, even in the absence of application programs, and then use that elevated baseline as the zero point for reported utilization numbers. Therefore, in some cases thermal diagnostics must take into account that even though a low processor utilization of 5% or 10% may be reported, there are actually dozens of watts of heat being dissipated by the processors.

When processor utilization data is available, thermal diagnostics uses the information to narrow the gap between $W_{\min}$ and $W_{\max}$ for the associated components. Since temperature rise is directly related to wattage dissipation, the net effect of constraining the expected wattage dissipation is to reduce the range of expected temperatures for these subsystems. The narrower temperature range means that fewer modeled scenarios are likely to match the actual temperature readings; only the best-fit scenarios produce low figure-of-error values as a range of expected temperature values is tightened.

Ultimately, thermal diagnostics works best if the power being dissipated by each major subsystem can be measured by onboard instrumentation that is independent of the operating system or hypervisor. Such power-measuring circuits need not be particularly precise in order to be valuable for thermal diagnostics. Thermal diagnostics promises to provide its most accurate results in the presence of power-measuring instrumentation that is capable of individually monitoring all major heat-dissipating subsystems, such as processors and dual in-line memory modules (DIMMs).

Partial obstructions and overheating

Some problem scenarios such as vent obstructions and component overheating are not binary; that is, they are not simply present or absent. Instead, some problems

may be present in varying degrees of severity. Most modeled problem scenarios therefore use incremental steps to cover varying extent. For example, an obstruction scenario may be modeled as a high value of impedance in one location, corresponding to a total obstruction at that location. However, alternate scenarios can be created that use lower values of impedance at this same location, corresponding to a 25% obstruction, 50% obstruction, 75% obstruction, etc. The use of incremental steps for obstructions, overheating, and other failure modes makes the model more accurate by providing multiple narrow-temperature-range targets instead of one broad-temperature-range target.

Sensor errors

It is possible that the temperature readings produced by an individual temperature sensor within the enclosure may be erroneous. Erroneous sensor readings can themselves be very disruptive if not handled properly and may, for example, lead to incorrect decisions to shut down subsystems or increase fan speeds (with a corresponding increase in audible noise levels). Thermal diagnostics therefore deals with this situation by including scenarios that exactly match the actual hardware configuration except for the introduction of an error in the reading obtained from various sensors. For example, a temperature sensor that is producing values that are 5°C higher than true readings may be modeled by doing all of the normal FNM calculations described previously and then adding 5°C to the temperature computed in the affected subsystem. The model is generally able to differentiate these scenarios from others, such as an overheating component, because the temperatures are not elevated in other parts of the enclosure that are downstream of the affected subsystem. The erroneous sensor-reading scenarios can be modeled at all sensors, and various degrees of error (both high and low) can be modeled as described above.

Postmortem computations

The root cause of a problem is most useful when it is determined at the time of the problem so that it can influence the remedy applied to the system. However, thermal diagnostics methodology is also useful as a postmortem diagnostics tool. If a log file exists and the log file includes information on the hardware configuration, processor utilization data (or wattage consumption of subsystems), and temperature readings, thermal diagnostics can analyze the system using FNM techniques after the fact to determine whether a thermal crisis may have contributed to unexpected system behavior.

Preliminary laboratory results

A prototype of BladeCenter thermal diagnostics has been developed in C++ and Sun Java**. The prototype collects inventory information, temperature readings, and processor utilization metrics from an operational chassis and performs the thermal diagnostic analysis described here. The current prototype uses IBM Director to collect the information that is analyzed and to drive responses.

We succeeded in correctly calling out the operational state and ambient room temperature for the BladeCenter system when it was operating nominally, when it had one processor blade fully obstructed, when processor blade filler panels were removed, and when an erroneous temperature value was reported by the ambient temperature sensor in the media tray. The sensor failure was simulated by altering a reported temperature in one data file.

The performance of the algorithm is shown in **Figure 3**. Figure 3(a) shows the variation of a temperature on processor blade 14 and a temperature on processor blade 12 over time, under two fault conditions: processor blade 14 blocked and three removed filler panels. The temperature regions associated with various thermal diagnoses are also shown on the figure: nominal conditions, processor blade 14 blocked, and missing filler panels. Note that while we present a two-dimensional representation of the temperatures within the unit by showing temperatures from only two sensors, in reality there are dozens of temperature sensors, and therefore the diagnosis regions are not two-dimensional areas but rather *n*-dimensional spaces.

The system starts off with the temperatures of both sensors within the nominal region, approximately 35°C for each processor blade. When processor blade 14 is blocked, the temperature of the blade rises according to the time line shown in the figure, coming to a steady state within the region associated with blockage of processor blade 14 within 25 minutes after blockage. The temperature of processor blade 12 does not change appreciably for this particular fault scenario.

The figure also shows the temperature profile evolution associated with the removal of three filler panels (with the pop-up dampers disabled to enhance the air short circuit through the empty slots). In this fault scenario, the temperature of both blades increases as flow is diverted through the open bays, coming to a steady state within the region associated with missing filler panels within 14 minutes after panel removal.

Figure 3(b) shows the figure of error predicted by five hypotheses plotted against time after processor blade 14 is blocked. Recall that the thermal diagnostics algorithm chooses the hypothesis with the lowest figure of error as the most likely diagnosis. The separation between the figures of error of different hypotheses determines the accuracy and confidence with which a diagnosis can be made. This figure also shows how soon a diagnosis can be made after the onset of this particular thermal anomaly.

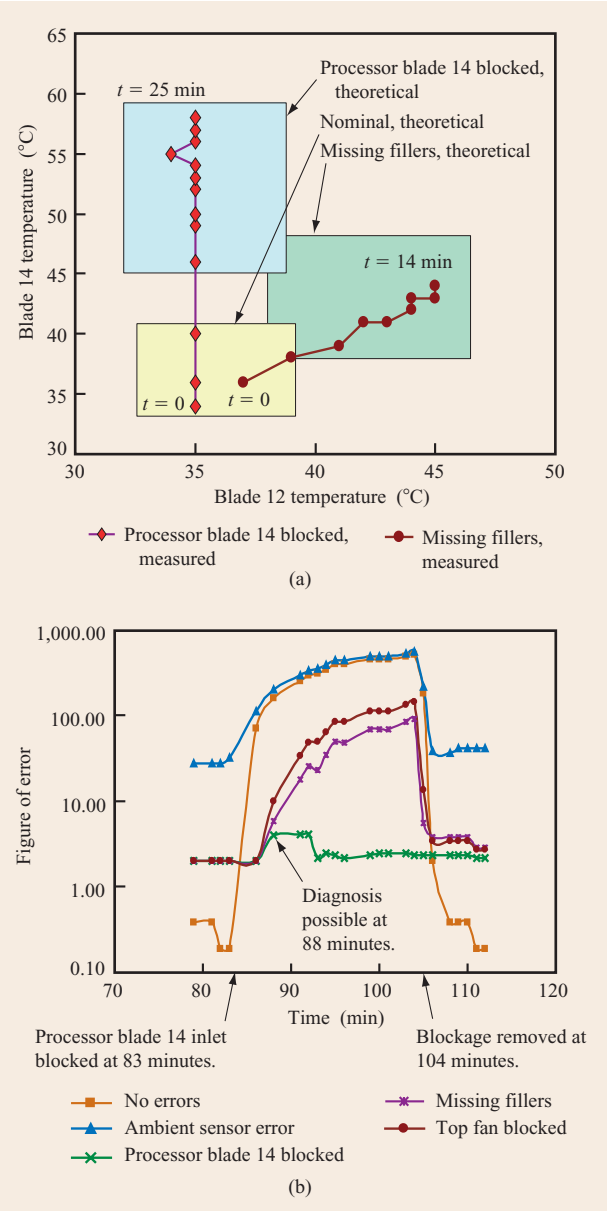

Figure 3

(a) Comparison of thermal diagnostic results for two selected problem scenarios. (b) Timescale of the diagnosis: Within two minutes of a processor blade obstruction being applied, thermal diagnostics can determine that conditions are no longer nominal; within five minutes it can give an unambiguous and accurate diagnosis.

The data shows that, at the time of the blockage at 83 minutes, the minimal figure of error corresponds to the nominal, fault-free scenario. After processor blade 14 is blocked, the figure of error associated with this scenario rises quickly; at 88 minutes, the figure of error corresponding to a blockage of processor blade 14 is lowest, and a confident diagnosis of this scenario is possible.

Although only five hypotheses are shown here, the algorithm tests literally tens of thousands of hypotheses before arriving at a final diagnosis. The program currently takes about 45 seconds to generate and test all of these scenarios on an IBM ThinkPad* T41. We are currently focusing on the accuracy of the analysis and have not made an effort to optimize the execution time. We expect that some improvements will be possible in this area.

In this and other scenarios, we have successfully demonstrated that thermal diagnostics can accurately call out the root cause of a thermal problem well before the BladeCenter chassis over-temperature mechanisms are invoked, thereby providing a predictive failure analysis capability for BladeCenter thermal issues.

Conclusion

This paper has presented a concept called *thermal diagnostics* that allows software to determine the root cause of thermal problems in electronic equipment. Using thermal diagnostics, it is possible to diagnose root causes that are outside the enclosure, such as an external obstruction, and those that involve a subsystem with no temperature-monitoring capability, such as removal of a side panel of the enclosure. The thermal diagnostic methodology is to simulate various types and degrees of thermal problems by repeatedly manipulating a flow network model of the equipment in an attempt to find the scenario or scenarios giving rise to temperatures that most closely match actual temperatures reported by the equipment. The analysis technique allows the equipment to invoke autonomic responses to certain situations and can result in quicker diagnosis and correction of thermal problems. Key portions of the algorithm have been modeled and verified to work well. Our model does not factor in radiated or conducted heat; this appears not to cause inaccurate results on current hardware. Future versions of the program could include those effects for more accurate predictions. Recommendations for future work also include further refinement and calibration of the models and algorithm (including the effects of altitude) and expanding the concept to non-BladeCenter equipment and higher-level entities, such as entire racks and data centers.

Acknowledgments

The authors thank Alex Cutting, Evan Huston, and Teresa Karr for their work in support of this project as part of a Spring 2005 Extreme Blue* project in Research Triangle Park, North Carolina. We also acknowledge the reviewers of this paper, whose comments were helpful in improving its quality.

*Trademark or registered trademark of International Business Machines Corporation.

**Trademark or registered trademark of Sun Microsystems, Inc. or Intel Corporation in the United States, other countries, or both.

References

1. D. J. Bradley, R. E. Harper, and S. W. Hunter, "Workload-Based Power Management for Parallel Computer Systems," *IBM J. Res. & Dev.* **47**, No. 5/6, 703–718 (September/November 2003).
2. V. Castelli, R. E. Harper, P. Heidelberger, S. W. Hunter, K. S. Trivedi, K. Vaidyanathan, and W. P. Zeggert, "Proactive Management of Software Aging," *IBM J. Res. & Dev.* **45**, No. 2, 311–332 (March 2001).
3. D. M. Desai, T. M. Bradicich, D. Champion, W. G. Holland, and B. M. Kreuz, "BladeCenter System Overview," *IBM J. Res. & Dev.* **49**, No. 6, 809–821 (2005, this issue).
4. G. Pruett, A. Abbondanzio, J. Bielski, T. D. Fadale, A. E. Merkin, Z. Rafalovich, L. A. Riedle, and J. W. Simpson, "BladeCenter Systems Management Software," *IBM J. Res. & Dev.* **49**, No. 6, 963–975 (2005, this issue).
5. M. J. Crippen, R. K. Alo, D. Champion, R. M. Clemo, C. M. Grosser, N. J. Gruendler, M. S. Mansuria, J. A. Matteson, M. S. Miller, and B. A. Trumbo, "BladeCenter Packaging, Power, and Cooling," *IBM J. Res. & Dev.* **49**, No. 6, 887–904 (2005, this issue).
6. G. N. Ellison, "Fan Cooled Enclosure Analysis Using a First Order Method," *Electron. Cooling* **1**, No. 2, 16–19 (October 1995).
7. A. Minichiello, "Flow Network Modeling: A Case Study in Expedient System Prototyping," *Proceedings of the International Conference on Thermal, Mechanics and Thermomechanical Phenomena in Electronic Systems,* 2000, pp. 70–89.

Received December 16, 2004; accepted for publication March 4, 2005; Internet publication October 7, 2005

William J. Piazza *IBM Systems and Technology Group, 3039 Cornwallis Road, Research Triangle Park, North Carolina 27709 (wpiazza@us.ibm.com).*

Mr. Piazza is a Senior Technical Staff Member in the xSeries Architecture and Technology Department. He received a B.S.E.E. degree in 1978 from the University of Pittsburgh and an M.S. degree in computer science from the University of Miami in 1996. He joined IBM in 1978 in Boca Raton, Florida, and has worked on the firmware, diagnostics, management software, and architectural design of several IBM desktop, server, and retail store systems. Mr. Piazza represents IBM on the PCI SIG Hot-Plug workgroup; he is the author or coauthor of numerous patents.

Richard E. Harper *IBM Research Division, Thomas J. Watson Research Center, P.O. Box 218, Yorktown Heights, New York 10598 (reharper@us.ibm.com).*

Dr. Harper is a Research Staff Member in the Server Technologies Department at the IBM Thomas J. Watson Research Center. He received a B.S. degree in physics and an M.S. degree in aerospace engineering from Mississippi State University in 1976 and 1977, respectively, and a Ph.D. degree in computer systems architecture from the Massachusetts Institute of Technology in 1987. He joined IBM in 1998 and has worked on architecture and performance analysis of high-end nonuniform memory access/symmetric multiprocessing system (NUMA/SMP) architectures, proactive problem prediction and avoidance technologies, and workload and power management. Dr. Harper is the author or coauthor of numerous patents and technical papers, and has supervised more than two dozen graduate student thesis projects.

Martin J. Crippen *IBM Systems and Technology Group, 3039 Cornwallis Road, Research Triangle Park, North Carolina 27709 (crippen@us.ibm.com).*

Mr. Crippen is a Senior Technical Staff Member in IBM eServer Power, Packaging, and Cooling Development. He received a B.S. degree in mechanical engineering from the Rochester Institute of Technology in 1983 and an M.S. degree in mechanical engineering from Binghamton University in 1994. His experience ranges from the development of mainframe computers to open-system VME computers. Mr. Crippen's field of specialty is electronics packaging, including card-on-board and server-level and rack-level packaging. He is well versed in system cooling and power system architecture. He has been instrumental in the development of many IBM Netfinity* and xSeries servers and is the lead mechanical engineer for the IBM eServer BladeCenter system. Mr. Crippen holds numerous patents in electronics packaging and cooling.

Jason A. Matteson *IBM Systems and Technology Group, 3039 Cornwallis Road, Research Triangle Park, North Carolina 27709 (mattesj@us.ibm.com).*

Mr. Matteson received an A.S. degree in engineering science from the State University of New York at Alfred in 1994 and a B.S. degree in mechanical engineering from the Rochester Institute of Technology in 1997. He began his professional career with IBM as a mechanical engineer, supporting the mechanical development team with system and hardware design. He received an IBM Outstanding Technical Achievement Award for the thermal design and support of one of the first 1U, dual-P4 Intel Xeon** servers. He supports IBM key customers with data-center cooling concerns and issues resulting from ultra-dense server deployments. Mr. Matteson has authored or coauthored several patents and technical bulletin disclosure publications.