RC 21843 (98313) 28 September 2000 **Computer Science**

IBM Research Report

Event Relationship Networks: A Framework for Action Oriented Analysis in Event Management

David Thoenen
IBM Advanced Technical Support
Raleigh, NC

Jim Riosa
IBM/ISM
Markham, Ontario
Canada

Joseph L. Hellerstein
IBM T. J. Watson Research Center
P. O. Box 218
Hawthorne, NY 10598

IFIP/IEEE
INTERNATIONAL SYMPOSIUM ON INTEGRATED NETWORK MANAGEMENT

Event Relationship Networks:

A Framework for Action Oriented Analysis In Event Management

David Thoenen (1), Jim Riosa (2), and Joseph L. Hellerstein (3)
(1) IBM Advanced Technical Support, Raleigh, NC, USA
(2) IBM/ISM, Markham, Ontario, Canada
(3) IBM T.J. Watson Research Center, Hawthorne, NY, USA

Abstract

Event management is a corner stone of high quality service delivery. To date, the focus of event management has been root cause analysis (RCA). We believe that this focus is misdirected in that it does not consider what action to take. Indeed, the root cause may not be actionable, and actions may be required for factors that are unrelated to a root cause (e.g., a condition that signals the end of a problem). We propose a new framework for event management--action oriented analysis (AOA). AOA assigns roles to events so as to determine the actions to take if an event is received. In many cases, the same event can have different roles depending on context. These ambiguities are resolved by analyzing event relationship networks. AOA also addresses what information is needed from event sources and subject matter experts. This information is identified and extracted by a set of four activities that we refer to as event management design. AOA has been used with much success at IBM's ISM installation and over fifty other production sites.

1. Introduction

The intense competition among service providers to demonstrate high quality service management (e.g., low response times, high availability) has led to very aggressive goals for IT-based services. Realizing these goals requires proactive management to provide early detection and isolation of problems. Traditionally, the role of event management in achieving these goals has been root cause analysis (RCA). However, it is our belief that proactive management requires a new approach to event management, one that focuses on the actions to take rather than analyzing the root cause of incidents. Herein, we describe an action oriented framework for event management that has been applied to production installations at ISM Canada and over fifty other large installations.

We begin with some definitions. A problem is one or more factors that affect service quality. An incident is an instance of a problem. Incidents have a starting and ending time. An event is a message that is generated, such as an SNMP trap. Events may precede, be concurrent with, and/or follow an incident.

Current work in event management focuses on RCA. [Finkel et al, 94] describe algorithms for identifying causes of event storms. [Houck et al 95] discusses operational experiences with RCA. [Lo and Chen, 99] propose an approach to RCA based on set algebra. [Meira and Nogueira, 00] describe how the computational demands of RCA can be reduced by using a recursive solution. [Kato and Koheilika, 99] address sequencing considerations for RCA. [Jakobson and Weissmann 93] describe the application of an expert system shell for telecommunication network alarm correlation. [Kliger et al 94] describe a code book based algorithm for RCA. [Liu et al., 99] describe ways to deal with complex, composite events in RCA. [Hasan et al., 99] present a unifying framework for RCA. Many of these technologies have been incorporated into event management products. For example: (a) Systems Management Arts (SMARTS) has packaged their codebook event correlation technology in the InCharge product (http://www.smarts.com); (b) Hewlett Packard delivers the correlation circuits of its Event Correlation Services in a variety of OpenView management products (http://www.openview.com); (c) NerveCenter, available from Veritas Software, employs finite state machines to model known behaviors (http://www.veritas.com); and (d) Tivoli Systems bases its enterprise event correlation facility, the Tivoli Enterprise Console, upon a Prolog rules engine (http://www.tivoli.com). Many of these products *could* be used to initiate actions when special conditions occur. For example, NerveCenter's state machine could initiate actions on state changes, and Tivoli's Prolog rules provide a complete programming environment. We see that throughout the focus is the same--root cause analysis.

Our thesis is that the foregoing efforts fail to meet the needs of proactive management. Proactive management requires rapid detection of and a fast response to exceptional situations so that the duration of an incident is shortened and its scope (number of systems or people affected) is narrower. Once containment is accomplished, a more detailed (and time consuming) analysis can be done to determine the root cause and to select corrective actions. Our view is that event management should provide the rapid response, and problem management should address the more detailed analysis.

Based on this, we claim that *root cause analysis is of secondary importance to effective event management*. We say this for several reasons:

1. Discovering the root cause is irrelevant to event management if no action can be taken in real time.

2. RCA cannot function unless the right set of events are provided, and determining the required set of events is beyond the scope of RCA (or at least has not been addressed in existing research).

3. RCA focuses only on the start of an incident. However, actions are also required as incidents transition (e.g., change in severity) and when an incident is resolved (e.g., closing a trouble ticket). Such considerations are not addressed by RCA.

4. RCA emphasizes the *root* of the causal chain. However, actions may be associated with symptoms as well. For example, a high temperature on a router necessitates a power shutdown. This action is taken whether the cause is a faulty fan or a defective power supply.

The foregoing motivates **action oriented analysis (AOA),** a new approach to event management that focuses on the actions to take. AOA encompasses the following: (1) identifying sources of events; (2) inventorying the events that can be generated and determining their meaning in the context that they are generated; (3) assigning roles to events depending on the problem context; and (4) developing correlation rules that identify event roles and initiate appropriate actions.

The remainder of this paper describes a technique that has been used with success by IBM consultants at over fifty customer installations. Section 2 describes event relationship networks (ERNs), the conceptual foundation of the approach that we propose. Section 3 discusses our methodology for implementing event management. Section 4 presents a case study of using ERNs at a large installation. Our conclusions are contained in Section 5.

2. Event Relationship Networks (ERNs)

This section introduces the key elements of event relationship networks and provides examples of their usage.

To motivate what follows, consider the case of a petroleum company that recently completed an event management implementation. The project was guided by two principles:

1. An automated electronic notification must be issued within seconds of the reception of any significant event by the enterprise event management software.

2. 100% of these notifications must be directed to the technical support organization with primary responsibility for the response to and recovery from the incident.

Analysis revealed that the company's current infrastructure was capable of generating over a thousand unique SNMP traps, which was later reduced to a core set of 450 events. Of these, many have an obvious association with an action (e.g., "printer jam"). On the other hand, there are a substantial number of events that act as a primary trigger for a response action in some situations and, under other circumstances, only provide secondary indications. The challenge facing the design team was to identify these dual-role events and to describe the logic required for assessing their role.

This challenge is best understood through an example. Consider a Cisco router. The environmental monitoring subagent checks for adverse temperature changes. Three temperature thresholds are used: warning, critical, and dangerous. Each threshold is associated with an event that is generated when that threshold is crossed. Further, if the "dangerous threshold" is exceeded, the router sends a "shutdown" event and powers-down to protect its circuitry. At any point when the temperature drops below the "dangerous threshold", a "temperature normal" notification is emitted, if the router has not powered down.

At first glance, it seems that the ideal response scenario is as follows. Upon receipt of the "temperature warning" trap from the router, a trouble ticket is opened and a beeper notification is issued to the router support organization. When the "temperature critical" notification is received, it is recognized as related to the trouble ticket previously opened, and the information contained in this event is appended to the existing trouble ticket. If a "shutdown" notification is received, this is appended to the trouble ticket.

This seemingly simple scenario is more complex in practice. First, the “temperature warning” notification is not issued in all cases. Should the temperature rise quickly, the “temperature critical” notification may be the first event issued. In extreme cases, only the “shutdown” event is issued. Thus, when a “temperature critical” is received, it is necessary to determine if a prior “temperature warning” has already been processed and a trouble ticket has been opened. If “temperature critical” is the first indication of the incident, the system must recognize that a trouble ticket has not been opened. Similar considerations apply to the “temperature critical” event.

Note that in the foregoing there is no root cause analysis. Rather, the goal is determining the correct action to take. Doing so requires considering events within the context of other events.

We now introduce a key concept: **event roles**. An event plays a primary role (is a **primary event**) if it provides an immediate, often unambiguous, indication as to the corrective action to take. For example, if a warning trap is the first event in the sequence above, then it is a primary event. Proactive management uses the receipt of a primary event to trigger a first level of response.

An event plays a secondary role (is a **secondary event)** if it is always extraneous in terms of selecting the corrective action in an exceptional situation. Although secondary events do not affect the choice of correct action, they may invoke actions of their own. For example, consider the above scenario and assume that the “temperature warning” trap is always generated. Then, “temperature critical” is a secondary event. Associated with this event may be an update to the trouble ticket.

If events were always either primary or secondary, then correlation would be much less complex. However, in a large number of cases, the role of an event depends on context. Further, and more importantly, the action associated with an event is often determined by its role. For example, consider the action of opening up a trouble ticket in the above scenario, which depends on whether the event is a primary or a secondary. Events that may be either a primary or a secondary are called **primary/secondary events**.

There is a fourth kind of event role---events that identify the end of an incident. We refer to these as **clearing events**. An example of such an event is “temperature normal”. Clearing events also indicate actions to take, such as closing a trouble ticket. We note in passing that the concept of a clearing event is typically foreign to technologies focused on root cause analysis. In fact, existing approaches to RCA that address clearing events rarely consider what actions to. In contrast, AOA considers clearing events to be central to event correlation.

Our approach to describing incidents uses a conceptual framework called **Event Relationship Networks (ERNs).** An ERN is a directed acyclic graph. Nodes are events and are labeled with one of the roles just described. Arcs from clearing to primary events indicate how an incident initiated by that primary event can be terminated. Arcs from primary to secondary events (or primary/secondary) indicate that the latter is associated with the former. We emphasize that these are not causal relationships, nor are they temporal relationships.

As indicated earlier, our focus is on the action to take. To this end, actions are associated with events based on their role in an incident. Typically, primary events initiate a response (e.g., opening a

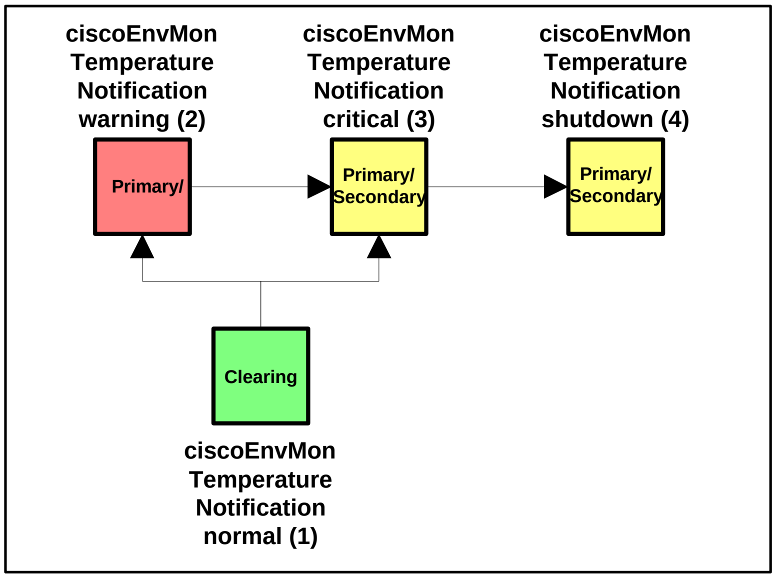
ciscoEnvMon Temperature Notification warning (2)
ciscoEnvMon Temperature Notification critical (3)
ciscoEnvMon Temperature Notification shutdown (4)
Primary/
Primary/ Secondary
Primary/ Secondary
Clearing
ciscoEnvMon Temperature Notification normal (1)

Figure 1: ERN for Cisco Temperature Notification

trouble ticket), and clearing events terminate the response (e.g., closing a trouble ticket). Secondary events, if they have associated actions, complement what the primary event has done (e.g., append to the trouble ticket).

A simple event relationship network for a Cisco router is shown in Figure 1. The Cisco EnvMon subagent emits temperature warning, critical, and shutdown events. The arrow from ciscoEnvMonTemperatureWarning to ciscoEnvMonTemperatureCritical depicts the relationships between these events. Note that, in general, there may be many secondary events for a single primary and many primary events for a single secondary.

Once an ERN is constructed, it is relatively simple to create the associated correlation rules. Consider the ERN in Figure 1. Here, we have the following rules:

```
Rule for [Notification_Warning]
 Open Trouble Ticket
End

Rule for [Notification_Critical]
 If [Notification_Warning] Received in Prior [Ten
Minutes],
      Then Execute Secondary Action
 Else Execute Primary Action.
      Action:  Primary
           Open Trouble Ticket
      Action:  Secondary
           Attach Secondary Event Data to Trouble Ticket
End
```

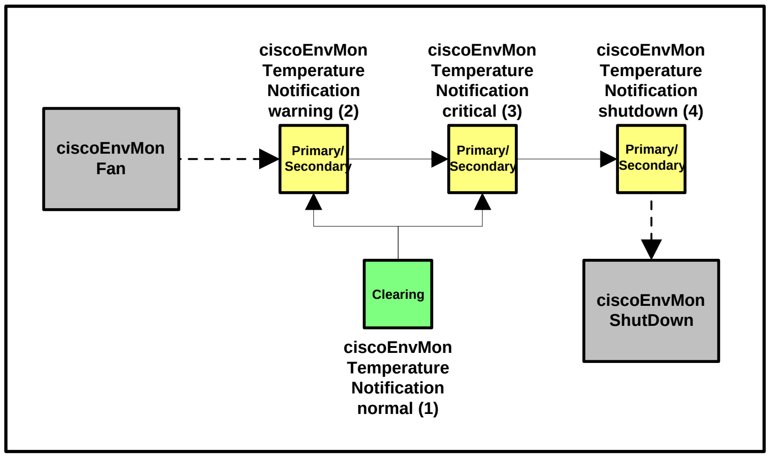
ciscoEnvMon Temperature Notification warning (2)
ciscoEnvMon Temperature Notification critical (3)
ciscoEnvMon Temperature Notification shutdown (4)
ciscoEnvMon Fan
Primary/ Secondary
Primary/ Secondary
Primary/ Secondary
Clearing
ciscoEnvMon ShutDown
ciscoEnvMon Temperature Notification normal (1)

Figure 2: Exended ERN for Cisco Temperature Notification

```
Rule for [Notification_Shutdown]
 If [Notification_Warning, Notification_Critical] Received
in Prior Ten Minutes,
      Then Execute Secondary Action
 Else Execute Primary Action.
      Action:  Primary
           Open Trouble Ticket
      Action:  Secondary
           Attach Secondary Data to Primary Trouble Ticket
End

Rule for [Notication_Normal]
 Attach Clearing Data to Primary Trouble Ticket
 Close [Notification_Warning, Notification_Critical]
 Close [Notification_Normal]
End
```

ERNs can grow to be quite complex. For this reason, it is desirable to provide nested structures of ERNs. For example, after some thought, it might be determined that ciscoEnvMonTemperatureWarning is not a primary event since this can be caused by a fan problem. However, the ways in which fan problems are produced is complex and so should be in a separate ERN subnetwork. Further, the manner in which a router shuts down is complex, suggesting that yet another subnetwork is needed. Figure 2 displays the revised ERN that includes these subnets.

The value of the ERN analysis is greatly enhanced if appropriate tools are available. For example, we have developed a Visio based authoring system for ERN construction. We have also developed a tool that takes ERNs as input and generates Prolog rules as output.

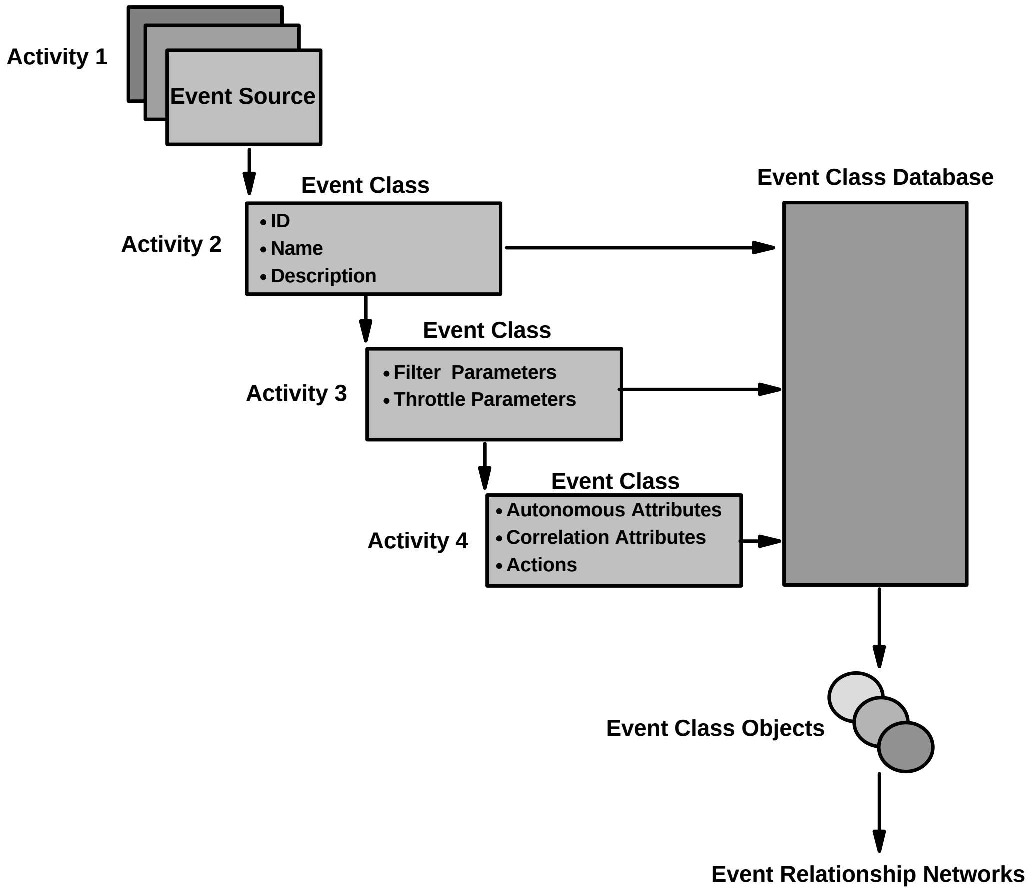

Figure 3: Flow of Activities in the Event Management Design (EMD) Methodology

3. Action Oriented Analysis Using Event Management Design

This section describes how we structure event management implementations. By event management implementation, we mean much more than selecting a technology for event correlation or even writing correlation rules. We mean tailoring the implementation to the specific needs of the installation. This entails determining: (a) which events should be filtered; (b) how events can be correlated; and (c) what actions should be taken.

Our approach is to structure the event management design project into four activities. The totality of these Activities constitute the **Event Management Design (EMD)** methodology. These are:

1. Select the event sources
2. Take inventory of all events
3. Document event policy and procesing decisions
4. Construct ERNs for correlation analysis

Figure 3 summarizes the EMD methodology, depicting the information content for each Activity. We discuss each in turn.

3.1 Select the Event Sources

In Activity 1, the designers review the IT environment and determine which managed resources (or event sources) will be considered within the scope of the design. Example of event sources include: UNIX servers, NT servers, NetWare Servers, hubs, routers, ATM switches, UPS systems, applications, web servers, and databases. In general, any IT resource or application that issues event messages (alerts, alarms, traps, console messages) in any format is a candidate event source.

The purpose of this activity is to provide an initial focus. Failing to include an event source means that the events it generates will not be incorporated into ERNs and hence actions based on these events will not be taken. On the other hand, including event sources unnecessarily makes later activities much more burdensome.

With these objectives in mind, the designers want to answer several questions that relate to the scope of the implementation. For example, is it only the mail servers, or should it include the entire infrastructure of a business unit? Also, along these lines, should the design focus upon the management requirements of a specific business service or a set of related services? Further, should the design concentrate upon the management of specific components, such as applications, databases, server hardware platforms and operating systems, or network components?

Typically, a design project considers between 10 and 25 event sources.

3.2 Take Inventory of Events

Activity 2 documents the events produced by each event source identified in Activity 1. This is accomplished by having a workbook (usually based upon a spreadsheet application) for each event source, with a description of events emitted by that source. Event descriptions include: the name of the event (e.g., “temperature warning” trap), the circumstances under which it is generated, the context information it contains (e.g., references to a Cisco host), and other factors that affect how the event is used in ERNs and action initiation.

Event source repertoires range from as few as one or two events to several thousand. For example, NT, Lotus Notes, and Nortel routers are all in the 4K range.

We should elaborate on the format of the workbook. Although it is contained in a spreadsheet, this is actually a highly customized application that employs extensive database capabilities to represent relationships between events. Indeed, our experience has been that effectively representing the information produced by Activity 2 requires considerable thought.

3.3 Document Event Policy and Processing Decisions

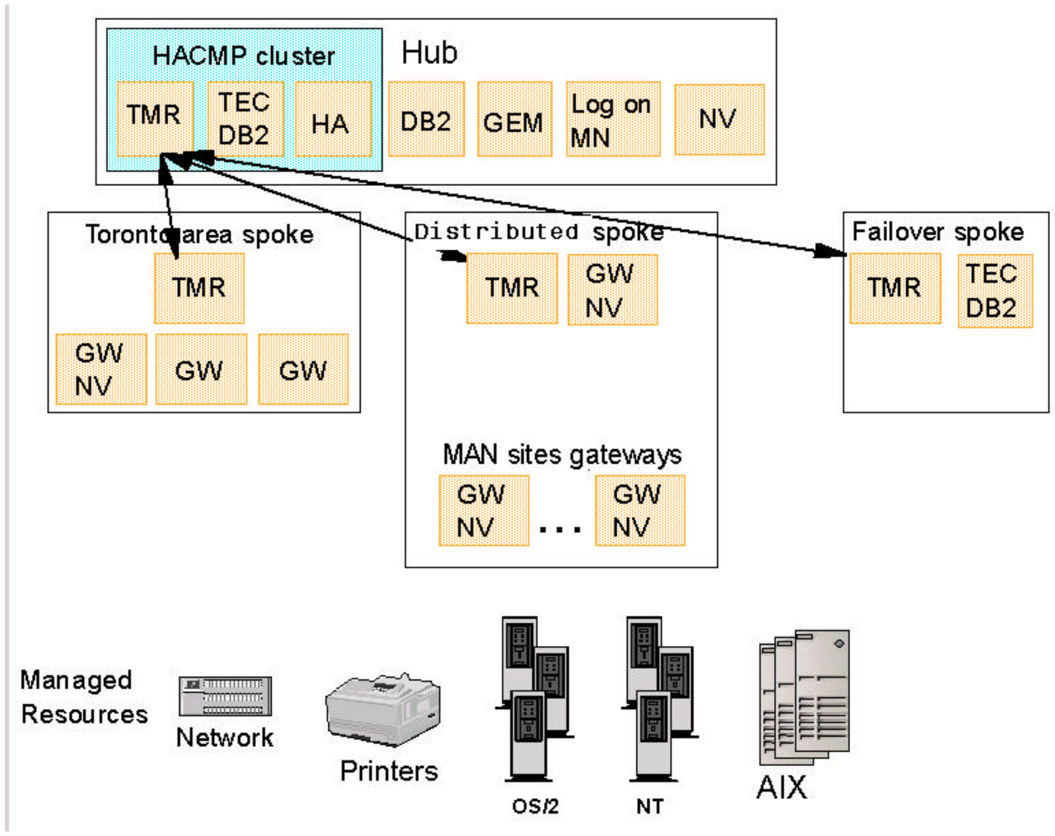

Figure 4: Structure of the ISM Installation in the Case Study

Activity 3 addresses how events should be processed. This is accomplished in two parts: policy specification and processing decisions.

The first step is to provide a framework for making decisions about the significance of events and the actions to take. We refer to these considerations as **event processing policies**. These policies guide the project through the decision making process regarding event filtering, forwarding, and correlation. Some of the issues addressed in this activity are: (a) What are the criteria for filtering events instead of forwarding then to the management station? (b) How will event severity levels be assigned? (c) How do event management severity levels relate to the severity levels defined within the problem management process for trouble ticket handling? (d) Where should less significant events be filtered? At the event source? At the event manager?

An example may help clarify the content of an event processing policy. A major Canadian bank has two key event processing policies. The first is that a trouble ticket must be opened for all primary events. The second is that secondary events are not discarded until the trouble ticket opened by the primary event is closed.

With the policy framework in hand, the next step is to make decisions about individual events. The starting point is the work product produced in Activity 2. This activity requires the involvement of **subject matter experts (SMEs)** who understand the event sources and their semantics. Typically, 75% of the event repertoire is considered to be of little significance and so are flagged for filtering from the event stream. However, events that emanate from lower parts of the OSI stack are more likely to be used than those from higher levels, in large part because the latter tend to be informational events rather than indicative of actions to take.

3.4 Construct ERNs for Correlation Analysis

Activity 4 focuses on events identified as “enterprise significant” in Activity 3. Such events are forwarded to the enterprise event manager.

Some enterprise significant events are used in ERNs to correlate with other events. For example in Figure 2, ciscoEnvMonTemperatujreNotificationWarning and ciscoEnvMonTemperatureNotificationCritical are correlated with one another. However, there are some events that are “autonomous” in that they are unrelated to other events and therefore are not candidates for event correlation. Actions for autonomous events may be initiated without writing correlation rules.

Our experience is that 50% of enterprise significant events are autonomous. Therefore, 12.5% of the total event repertoire requires correlation at the event manager.

For the remaining events, the “correlation candidates”, the events are analyzed and the correlation relationships which must be reflected in the event processing rules are documented in ERNs.

4. AOA Experience and Refinements

This section describes our experience with AOA at a complex installation managed by ISM Canada. We begin with a description of the environment and then summarize issues that arose in implementing ERNs. In particular, we discuss how AOA has been refined to automate the translation of ERNs into correlation rules using techniques developed at ISM.

Figure 4 describes the logical structure of the ISM site. There is a wide range of managed resources: printers, print servers (OS/2 machines), AIX (Unix) machines, SP2 (cluster) machines, and three critical business applications--Lotus Notes (email), ADSM (backup and restore), and printing. Management is organized into a hierarchy with a hub and spoke structure. Most servers are located in the Toronto spoke, which is the central site for Lotus Notes and ADSM. Print servers are distributed across the country and are accessed within various Metropolitan Area Networks (MANs). Tivoli management software is used throughout. This software is organized into regions, called Tivoli Management Regions

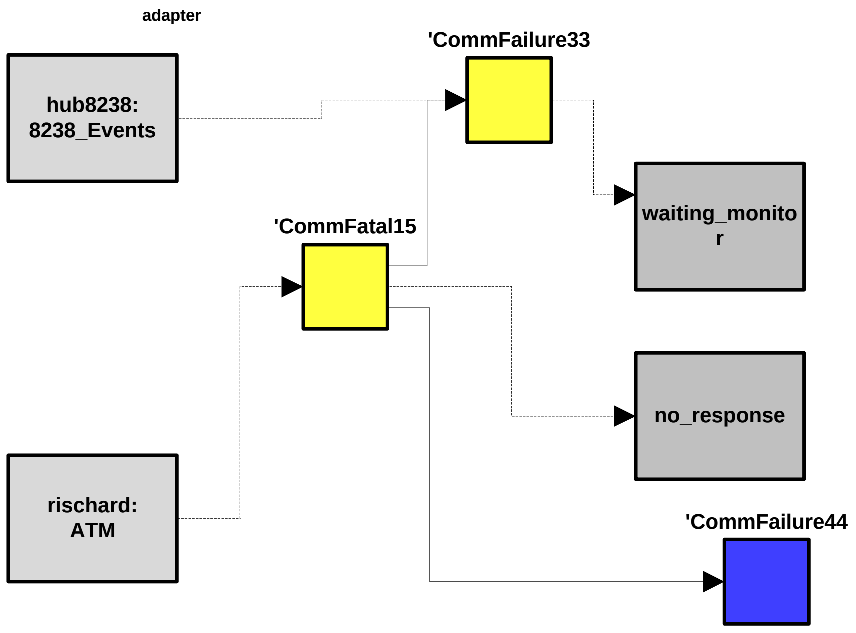

Figure 5: Adapter ERN Subnets

(TMR), with the Hub's TMR providing overall coordination. To enhance fault tolerance and scalability of the management stations, the Hub TMR runs on a cluster with fail-over capabilities (HACMP). Tivoli's event management console (TEC) uses a Prolog engine for correlation.

The event management implementation was undertaken with three objectives in mind: (1) automation, (2) standardization, and (3) adopt an end-user perspective. For (1), there are two kinds of automation. The first is automating the detection of exceptional situations and classifying the kind of situation present. ERNs played a central role in achieving this objective. The second kind of automation is taking actions. Here, ERNs provided a mechanism for determining the role of the events present, and the results of Activity 3 are used to determine the action to take.

For (2), the goal is to have a standard framework for event correlation that can be used in a wide range of installations. Such standardization is of tremendous importance to service providers in that correlations rules can be reused.

The implications of (3), adopting an end-user perspective, puts considerable burden on event correlation in that connections must be made between seemingly unrelated events. To illustrate this, consider the availability and performance of Lotus. Nearly all Notes servers are located in the Toronto server farm. Thus, end-user perception of Notes availability depends on several factors: network connectivity to the server farm, server load, and performance of the Notes application. As such, performance and availability considerations involve analysis of events not only from the application, but also from the supporting hardware, and the operating system that the application was running on.

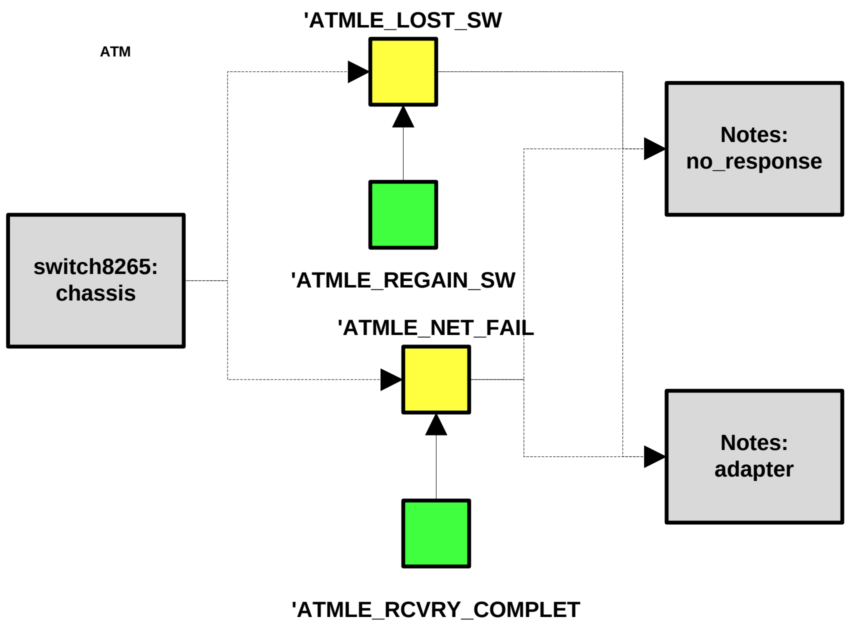

Figure 6: ATM ERN Subnets

Based on analysis of the event sources in Figure 4, we completed Activity 2 with a repertoire of 7,242 events. This was subsequently reduced to 376 events, which were the events that indicated an immediate or potential loss of a service. Out of the 376 events, 194 were identified as candidates for correlation. Additional events are included based on some 127 monitors available in the environment ERN construction proceeded along the lines of Activities 1-4 described in Section 3. We found that in a well structured management system, we can assign management strategies directly to primary events, secondary events, and clearing events. There are, however, some further subtleties. For example, a delay is often needed between the receipt of a primary/secondary event and taking an action. During this delay, we wait for additional events that may aid is in determining if an event is truly a primary or if it is actually a secondary. The length of this delay depends on the amount of degradation in availability and performance that can be tolerated in order to get finer grain problem identification.

We now provide details of a few ERNs used at ISM. Consider the events associated with adapters, as shown in Figure 5. The two ERN subnets to the right are Lotus events, while the ones on the left are events to be correlated from very different sources, an IBM 8238 hub and a RISC 6000 server.

What happens if a CommFatal15 is received? This is a primary/secondary event indicating a failed or busy adapter. Before notifying the Notes administrator, however, we want to determine if there is something happening on the server, rischard:ATM. Indeed, if server events are available, then CommFatal15 may be a secondary event. Otherwise, it is a primary.

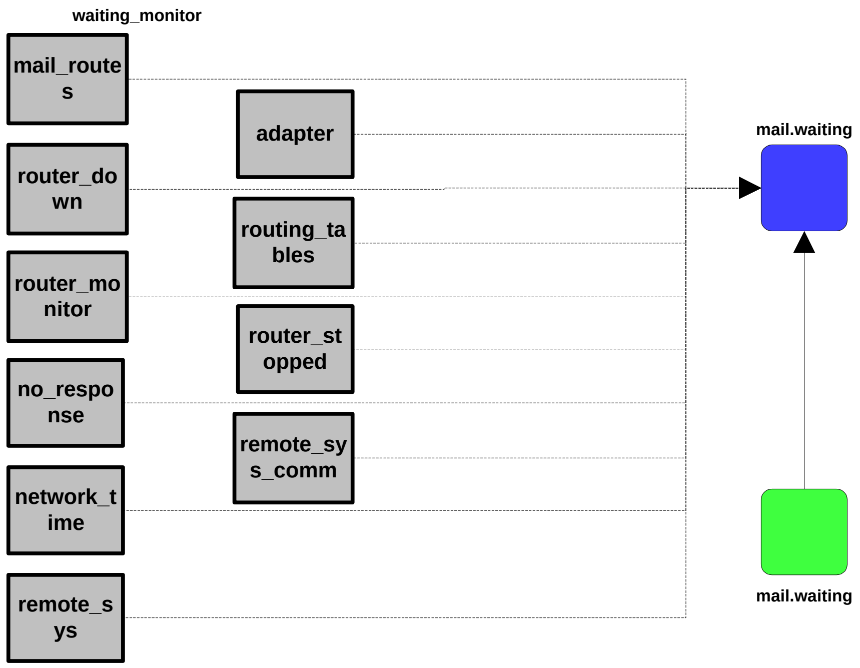

Figure 7: Waiting_Monitor ERN

To understand more about actions associated with rischard:ATM events, we examine the ATM ERN subnet. This is shown in Figure 6. Here, additional primary/secondary events are present, such as ATMLE_LOST_SW, which is a wire fault in the server. The actions associated with receiving such an event are to notify (a) hardware services personnel to initiate repairs and (b) Notes administrators that an ongoing performance problem is present for applications on the affected server.

Over time, we may see further effects of the ATM problem. Returning to Figure 5, we see that CommFatal15 affects is correlated with the waiting_monitor subnet. This ERN subnet is depicted in Figure 7. Note that the mail_waiting event is present within this subnet. Thus, if at some later time, this event is received and the incident is still open, we would know that the problem has been identified and an action has already been taken.

Once ERNs are constructed, they must then be translated into correlation rules. In our case, this means Prolog rules. This raises two questions. How is this translation accomplished in an automated way? And, once this translation is accomplished, how do we maintain and enhance the ERNs in a way that does not require manipulating Prolog rules?

We begin by briefly describing our approach to rule translation. In essence, we build a capability into the event management system that knows about the structure of ERNs and how to process them. Our approach is to develop a tool that generates Prolog rules that incorporate logic for processing ERNs.

ERN processing is based on ERNs being a directed, acyclic graph of events. Thus, the first step in our analysis is to provide a way to identify whether two events are on the same directed path within one or more ERN. This must be done in a manner that is computationally efficient. With this capability in hand, the receipt of an event *E* is handled as follows:

- If *E* is a primary/secondary event, a check is done to see if there is an primary/secondary or primary ancestor of *E*. If so, then *E* must be a secondary event. However, the absence of such an ancestor does not necessarily mean that *E* is a primary since events may not arrive in order of their position in a directed path. Thus, a delay is required before designating a primary/secondary event as primary.
- If *E* is a primary or a primary/secondary event, we check for descendants of *E*, that is, those events that are found on directed paths emanating from *E*. All such events are secondaries.

Note that pure primary and pure secondary events are handled easily by the above approach, although there is some subtlety in handling events that may be missing in a path. In addition, some thought is required as to how information is communicated about the events themselves. Our approach uses “slots” (fields) in the events. By properly encoding information about the ERN paths in which an event lies and placing this information into event slots, we can greatly reduce the number of complexity of Prolog rules generated by ERN translation. Last, note that the above scheme allows us to think in terms of ERNs, both for initial ERN construction and when modifications and extensions are done. This makes it much easier to use AOA in practice.

We now discuss the event processing policy developed at ISM. Our approach is as follows.

- Primary events: Immediately notify an appropriate administrator and open a trouble ticket.
- Secondary events: Provide an advisory notification, such as by e-mail or pop-up display on an administrators console. Also, append to the already open trouble ticket.
- Clearing events: Suspend all action on the primary and associated secondary events; advise the administrator that the operating status has returned to normal; and close the trouble ticket.

The modularity afforded by the above structure has served us well. For example, three months after the initial event management implementation, extensive modifications were applied to the ERNs so that actions could be based on two additional factors: severity (critical, minor, warning, harmless) and host that originated the event. Further, several new event sources were added (e.g., a database application and new servers). Although these modifications were extensive, they were easily incorporated into the ERNs and translated into Prolog rules.

The end result is an event management system that is driven by ERNs rather than Prolog rules. This has substantial benefits. Foremost, it provides us with a way to standardize event management in terms of ERNs. In addition, since ERNs provide a more natural representation of event management logic, we can work at a higher-level of abstraction and so avoid distractions and confusions caused by details of rule-based programming.

Before closing this section, we comment on action oriented analysis. AOA provides an immediate response based on events whose role has been determined through the ERN logic. Our experience has been that service is restored more rapidly using this approach and that doing so requires fewer participants. We continue to do root cause analysis. However, this is done as part of a detailed analysis of the incident.

5. Conclusions

Effective event management is a corner stone of delivering high quality IT services. Over the last decade, much effort has been expended to develop correlation technologies that provide better root cause analysis (RCA) for event management. It is our belief that these efforts are misdirected. Instead of focusing on RCA, it is much more effective to focus on the actions to take. This thesis motivates our development of a new framework for event management--action oriented analysis (AOA).

AOA views events as having one of four roles: primary events that specify an immediate action to take; secondary events that augment the information provided by primary events; clearing events that terminate an incident; and primary/secondary events that may be either a primary or secondary event. Events are related to one another in an event relationship network (ERN). ERNs are used to disambiguate primary/secondary events and to identify the specific problem incident so that actions can be taken.

AOA focuses on the actions that can be taken based on the information available. This information is determined by the event management design (EMD) methodology. EMD consists of four Activities: (1) selecting event sources, (2) inventorying the invents that are likely to be produced by these sources, (3) developing an event processing policy and applying this policy to specific events, and (4) constructing ERNs for correlation analysis.

Extensive experience with AOA has been obtained at the IBM ISM site. Initially, some adaptation was required to automate the translation of ERNs into an executable form (in our case Prolog rules). We did this by viewing ERNs as a directed acyclic graph and developing a way to encode into event slots (fields) information about the paths of the graph in which the event lies.

Our experience with ERNs has been excellent. In particular, at ISM, ERNs have resulted in event management that restores service more rapidly and requires fewer participants. We attribute this to ERNs providing a superb abstraction for representing and implementing action-oriented event processing.

References

Finkel, A., Houck, K., Calo, S., Bouloutas, A., ***An Alarm Correlation System for Heterogeneous Networks***, Network Management and Control, Vol. 2, Plennum Press, 1994.

Hasan, M., Sugla, B., and Viswanathan, R., ***A Conceptual Framework for Network Management Event Correlation and Filtering Systems***, International Symposium on Integrated Network Management, Boston, MA, May, 1999.

Houck, K., Calo, S., Finkel, A., ***Towards a Practical Alarm Correlation System***, Fourth International Symposium on Integrated Network Management, Santa Barbara, CA, May, 1995.

Jakobson, G., Weihmayer, R., Weissman, M., ***A Domain-Oriented Expert System Shell for Telecommunications Network Alarm Correlation***, Network Management and Control, Vol. 2, Plennum Press, 1994.

Kato, NeiOhta and Tomohiro KoheiIka, **Proposal of Event Correlation for Distributed Network Fault Management and Its Evaluation**, IEICE Transactions on Communications, Vol E92-B, number 6, 1999.

Kliger, S., Yemini, S., Yemini, Y., Ohlse, D, Stolfo, S., ***A Coding Approach to Event Correlation***, Fourth International Symposium on Integrated Network Management, Santa Barbara, CA, May, 1995.

Lewis, L., ***A Case-based Reasoning Approach to the Resolution of Faults in Communications Networks,*** in Proceedings Third International Symposium on Integrated Network Management, North Holland, 1993.

Liu, G., Mok, A.K., and E.J. Yang., ***Composite Events for Network Event Correlation***, International Symposium on Integrated Network Management, Boston, MA, May, 1999

Lo, C. C. and S. H. Chen, **Robust Event Correlation Scheme for Fault Identification in Communication Networks**, International Journal of Communications Systems, Vol. 12, No. 3, June, 1999.

Meira, D.M. and J.M.S. Nogueira, **A Recursive Approach for Alarm Correlation in Telecommunication Networks**, IEEE/IFIP Network Operations and Management Symposium, 2000.