Global Headquarters: 5 Speen Street Framingham, MA 01701 USA P.508.872.8200 F.508.935.4015 www.idc.com

WHITE PAPER

Invariant Analyzer: An Automated Approach to System Performance Management

Sponsored by: NEC

Tim Grieser

October 2011

INTRODUCTION: GROWING IT COMPLEXITY

Today's highly competitive business environments demand high availability and fast performance for critical business applications under widely varying operating conditions. Many applications are directly accessed by end users, over the Web, using a variety of portable and handheld devices. This places increasing pressure on IT organizations to manage applications and IT infrastructure to provide fast performance, avoid slowdowns, increase application availability, and prevent outages. Satisfying these requirements is made increasingly difficult by scale-out, multitier application architectures and deployments. Growing infrastructure complexity, including virtualization, mobility, and cloud architectures, is making the tasks of ensuring fast performance and achieving required service levels even more difficult.

This IDC White Paper addresses the growing requirements for IT performance management in today's complex environments and introduces Invariant Analyzer, an analytic engine from NEC for identifying and helping resolve performance problems in large, complex IT infrastructures.

KEY IT BUSINESS CONCERNS

The worldwide economic uncertainty is generating greatly increased pressures on IT organizations to achieve cost savings and operational efficiencies. Ongoing IDC surveys and interviews with IT managers, professional staff, and end users provide insights into key IT pain points and priorities. Typical IT business concerns include the following:

- Contain costs
- Improve customer satisfaction
- Improve quality and accuracy
- Increase revenue
- Speed time to market
- Increase market share

IT organizations must provide effective and responsive support to the business and to other organizational units by operating key applications with high service quality, especially for application and database performance and availability. At the same time, IT must operate efficiently so as to contain costs. This puts pressure on IT staff to prevent operational problems and to more quickly resolve problems when they occur.

PERFORMANCE MANAGEMENT OBJECTIVES

Performance management is an essential component of service-level management to ensure that performance and availability service objectives are achieved for key applications and workloads. Performance management enables IT organizations to track and report operational performance to understand the performance being experienced by transactions and end users, prevent slowdowns and outages, troubleshoot performance problems, identify root causes, eliminate bottlenecks, and meet service-level agreements (SLAs) to support business application requirements.

Key Foundation: Performance Monitoring

Performance management is founded on monitoring — the process of measuring, reporting, and tracking key performance metrics for a variety of infrastructure components and across multiple layers in the infrastructure and application stack. Typical components being monitored include:

- IT infrastructure (servers, storage, networks)
- Physical and virtual devices
- Applications
- IT and business services
- Synthetic and real transactions

Monitoring is typically performed by software that can take a variety of forms, including monitoring facilities built into operating systems (so-called "agent-less" monitors), software agents deployed on servers, network monitoring appliances that decode network traffic, and external points of presence that measure end-to-end transaction performance. Key metrics gathered by monitors include throughputs, transaction rates, utilizations, and response times, as well as device-specific measures such as memory consumption, paging and I/O rates, and device queue lengths to help identify bottlenecks.

Thresholds and Alerts

One of the key methods used for managing performance in operational systems is to set thresholds to detect when a performance metric, such as utilization, has reached a limiting value, such as 70%. The monitored metric is compared with the threshold value on an ongoing basis, and an alert, signifying an exceptional condition, is generated if the threshold is reached or exceeded. Essentially, alerts are a warning that a monitored metric has reached some critical value and that some action must be taken. In practice, thresholds are typically defined for a variety of performance metrics on a wide range of infrastructure devices, such as servers, storage, and networks, and for application and service metrics such as throughput and response time.

Threshold and Alert Challenges

While thresholds and alerts are a core technology for performance management, a number of challenges are associated with this approach. These challenges may be summarized as follows.

- **Setting, tailoring, and maintaining thresholds.** These tasks can require substantial IT staff time, especially for large-scale, complex infrastructures. Thresholds must be configured for large numbers of infrastructure elements and must be tailored to meet specific operational conditions — often an intensive manual effort. Thresholds need to be adapted to meet dynamic environments, such as daily/weekly/monthly variations or peak loads — one size does not fit all. Also, thresholds may not be defined for all the right performance metrics.

- **Tendency to produce "alert storms."** When many thresholds are defined and in operation, exceptional conditions can generate high volumes and bursts of alerts. This condition can flood the system with alerts, making it difficult to understand what the problem is.

- **Thresholds are typically for individual metrics** — such as CPU utilization. They do not show the relationship of an individual metric to other performance metrics — including SLA components such as response time.

- **Hard to determine root causes.** While thresholds and alerts are useful for signaling trouble conditions, they do not identify by themselves the source of problems or the root cause of the problems.

Overall, thresholds and alerts used in conjunction with monitoring provide a core technology for signaling that exceptional conditions are occurring that may be impacting the performance and availability of IT-based services.

INTRODUCING NEC INVARIANT ANALYZER

Because of the scale and the complexity of today's IT infrastructures, the task of identifying, analyzing, and resolving performance problems in operational systems is formidable. While monitoring can be used to identify many exceptional conditions, the job of identifying root causes is often left to individual IT analysts or even "war room" meetings of technology stack experts. IT analysts need powerful tools to simplify and extend their ability to discover and resolve performance problems in these dynamic, complex environments. NEC has introduced Invariant Analyzer, a behavior learning analytic engine for performance analysis, to address these needs. Invariant Analyzer is designed to proactively identify trouble spots, performance bottlenecks, and root causes.

Invariant Analyzer examines and analyzes large numbers of performance metrics across multiple infrastructure devices and across a wide variety of IT domains to discover and analyze performance problems. It works with standard monitor data from existing sources and incorporates a number of automated analysis capabilities. In contrast to a threshold approach, Invariant Analyzer discovers "normal" time-invariant relationships among performance variables collected during baseline operating time periods.

Invariant Analyzer discovers correlations between variables and constructs relational expressions connecting them, when appropriate. It has an extensive library of relationship formulas, including time series analysis and linear and nonlinear equations. Invariant Analyzer builds reference models of operational environments based on mathematical relationships, as well as templates and model prototypes. Relationships are tied to the target infrastructure components from which monitor data is recorded so that changes in relationships can be linked to specific infrastructure elements. In operation, Invariant Analyzer tracks ongoing operational relationships among the designated performance metrics and generates alerts when operational relationships deviate from the "normal" pattern that is predicted by the reference model. Such a deviation is known as a "broken invariant relationship" and signals that a problem condition is occurring.

Invariant Relationship Example

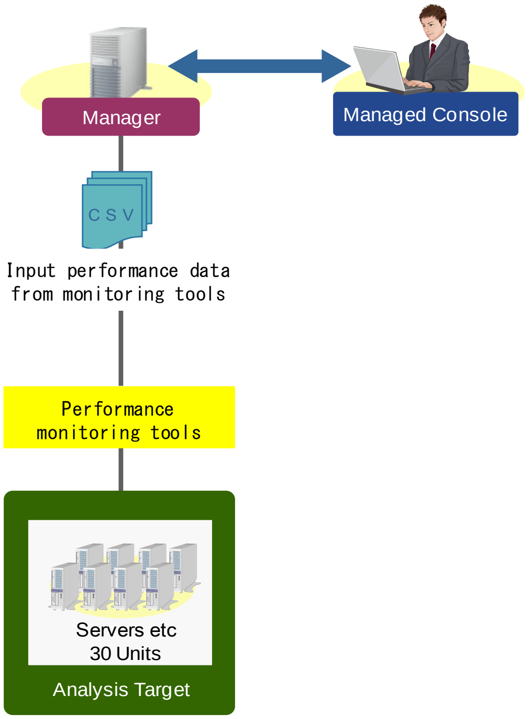
For example, a normal time-invariant relationship between two performance metrics such as transaction rate and response time might be linear under normal operating conditions. But as transaction volumes increase dramatically — such as during a peak period — some devices may become saturated and become bottlenecks, causing the relationship between response time and transaction rate to change and become nonlinear. This would be described as a "broken invariant relationship" because the linear relationship does not hold under peak loading conditions. The infrastructure elements associated with the performance variables in the broken relationship would be identified and would be strong candidates for analysis to determine the root cause of performance problems. Invariant Analyzer provides graphical displays and built-in analysis capabilities to support troubleshooting "broken invariants" and relate them to specific infrastructure elements and performance problems, as well as helping determine underlying root causes.

INVARIANT ANALYZER PRODUCT OVERVIEW

Invariant Analyzer consists of three major components. As shown in Figure 1, the Invariant Analyzer product configuration includes a user console (shown as Managed Console); a management server (Manager), which contains the analyzer and model software; and input capabilities for reading performance data gathered by standard monitoring tools from tracking IT infrastructure components that are targets for analysis.

FIGURE 1

Invariant Analyzer Overview


Manager
Managed Console
CSV
Input performance data from monitoring tools
Performance monitoring tools
Servers etc
30 Units
Analysis Target

Source: NEC, 2011

INVARIANT ANALYZER IN OPERATION

Invariant Analyzer is designed to meet the performance management challenge of how to analyze the enormous volume of monitoring data from numerous hardware devices, operating systems, and applications to identify performance problems before they impact users and to help identify root causes to guide solutions. In practice, Invariant Analyzer is used to implement a multistep process that can be summarized as follows:

- The target infrastructure devices are selected (example: 30 target servers).
- The key performance variables (counters) are identified (example: select 100 counters per server).
- Monitor data for the selected devices and counters is input by the Invariant Analyzer Manager.
- Initial multilevel analysis is performed on the input data to discover "invariant" relationships and build a baseline model. Baselines are built using a minimum of 100 intervals of monitor data — often far more.

- Invariant Analyzer tracks and analyzes monitor data from the operational system on an ongoing basis to determine whether any invariant relationships are being broken.

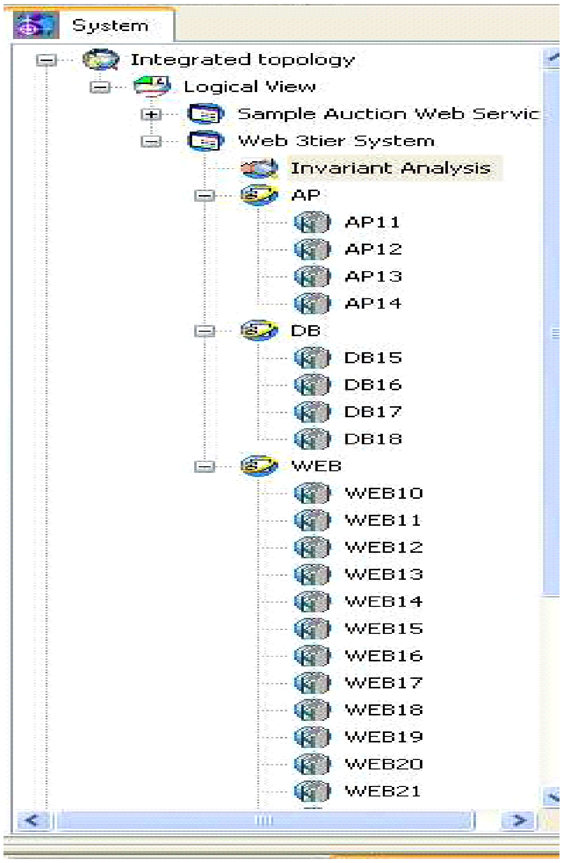
If relationships are being broken, Invariant Analyzer generates alerts and provides graphical analysis capabilities to aid in problem identification and resolution. Key Invariant Analyzer graphical reporting capabilities are described in the following sections.

Basic Screen Shows Analysis Targets

As shown in Figure 2, the basic Invariant Analyzer screen displays a hierarchical view of analysis targets. In this example, a three-tier Web-based system with application servers, Web servers, and database servers is being tracked and analyzed for potential broken invariant relationships or "anomalies."

FIGURE 2

Invariant Analysis Targets



System

- Integrated topology
  - Logical View
    - Sample Auction Web Servic
    - Web 3tier System
      - Invariant Analysis
      - AP
        - AP11
        - AP12
        - AP13
        - AP14
      - DB
        - DB15
        - DB16
        - DB17
        - DB18
      - WEB
        - WEB10
        - WEB11
        - WEB12
        - WEB13
        - WEB14
        - WEB15
        - WEB16
        - WEB17
        - WEB18
        - WEB19
        - WEB20
        - WEB21

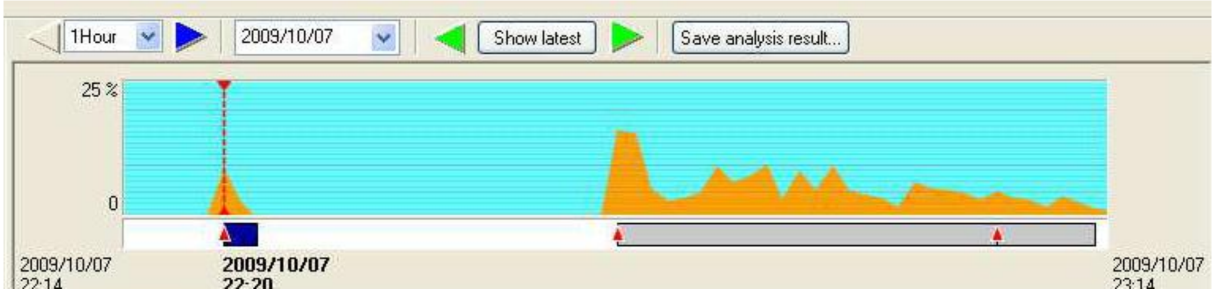
Source: NEC, 2011

Invariant Analyzer Anomaly Graph

Figure 3 illustrates the Invariant Analyzer Anomaly Graph. This report allows IT analysts to visualize failures graphically and indicates the time the failure occurred and the severity of the failure. The report shows graphically the percentage of broken invariant relationships relative to all invariant relationships in the model during an operating interval. Typically, 5% or higher is the mark for identifying that an anomaly has occurred and for Invariant Analyzer to generate an alert.

FIGURE 3

Invariant Analyzer Anomaly Graph



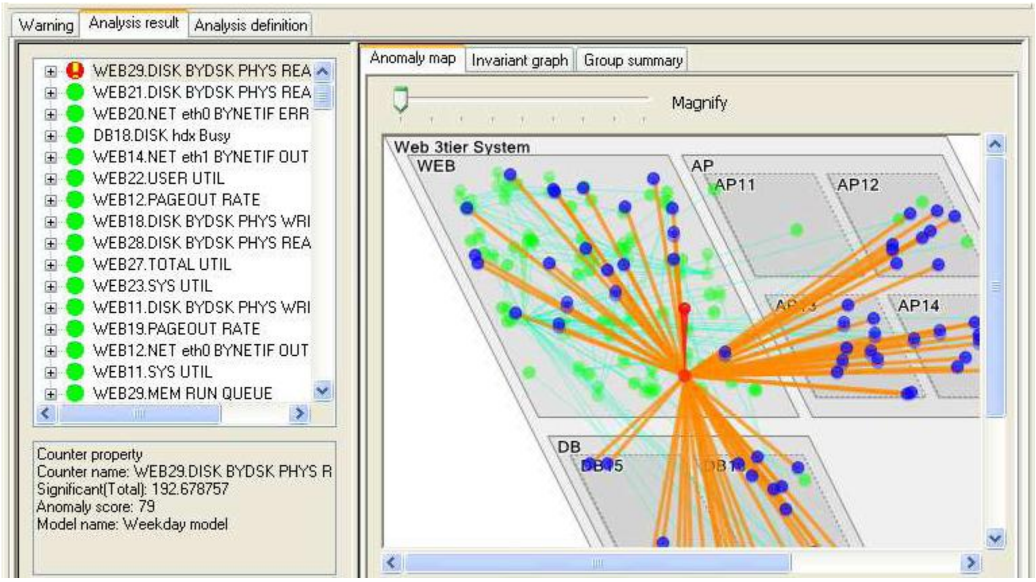
Source: NEC, 2011

Invariant Analyzer Anomaly Map

Figure 4 illustrates the Invariant Analyzer Anomaly Map. Invariant relationships tend to be broken in bunches or clusters rather than single failures. For each anomaly, the anomaly map identifies the primary metric associated with a whole set of broken invariant relationships. In this example, the infrastructure element WEB29.DISK is identified as the primary metric with the most broken relationships and is shown in the center of the graph. Broken invariant relationships are shown as orange lines in the anomaly map for the selected anomaly in the left panel.

FIGURE 4

Invariant Analyzer Anomaly Map



```
⊞ ● WEB29.DISK BYDSK PHYS REA
⊞ ● WEB21.DISK BYDSK PHYS REA
⊞ ● WEB20.NET eth0 BYNETIF ERR
⊞ ● DB18.DISK hdx Busy
⊞ ● WEB14.NET eth1 BYNETIF OUT
⊞ ● WEB22.USER UTIL
⊞ ● WEB12.PAGEOUT RATE
⊞ ● WEB18.DISK BYDSK PHYS WRI
⊞ ● WEB28.DISK BYDSK PHYS REA
⊞ ● WEB27.TOTAL UTIL
⊞ ● WEB23.SYS UTIL
⊞ ● WEB11.DISK BYDSK PHYS WRI
⊞ ● WEB19.PAGEOUT RATE
⊞ ● WEB12.NET eth0 BYNETIF OUT
⊞ ● WEB11.SYS UTIL
⊞ ● WEB29.MEM RUN QUEUE
```

```
Counter property
Counter name: WEB29.DISK BYDSK PHYS R
Significant(Total): 192.678757
Anomaly score: 79
Model name: Weekday model
```

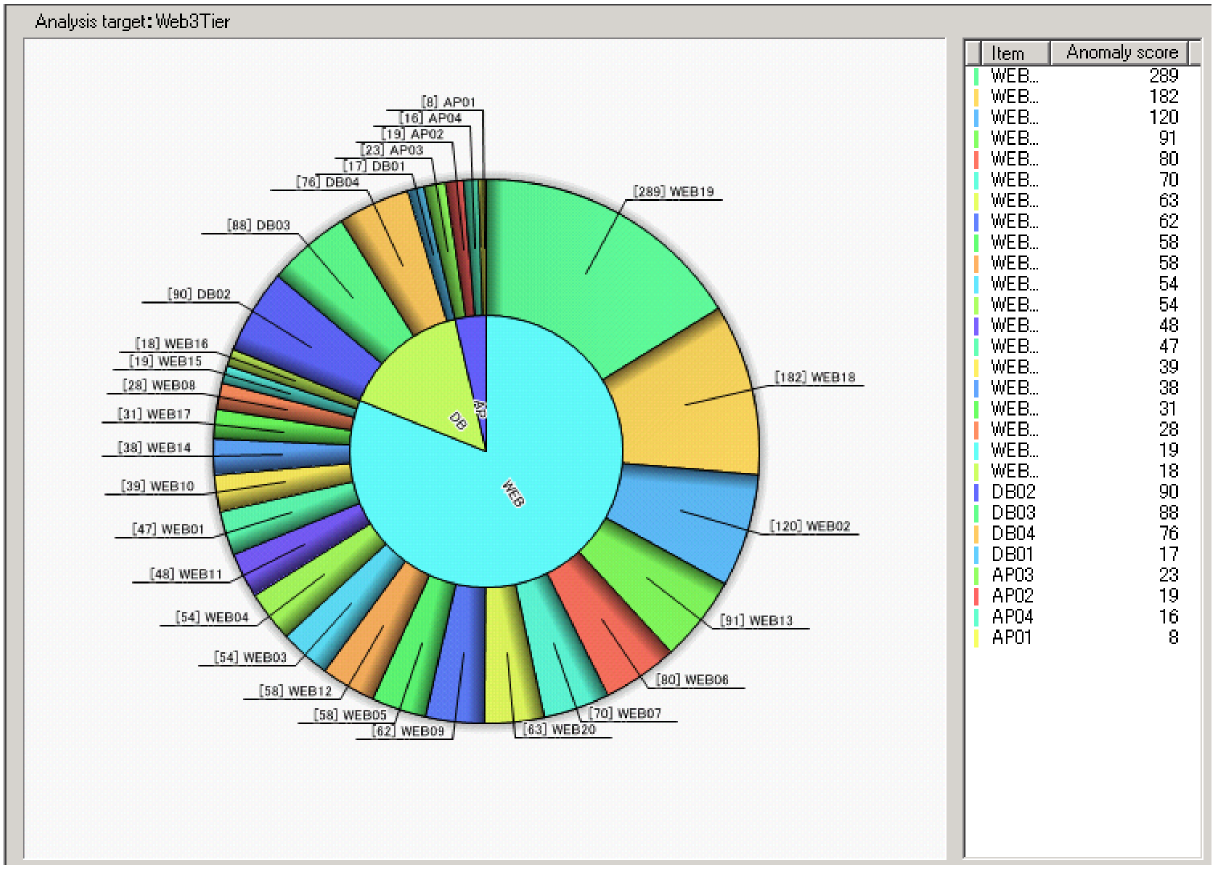
Source NEC, 2011

Invariant Analyzer Anomaly Pie Charts

Invariant Analyzer pie charts show the distribution of anomalies across components of the system to identify probable root causes. The pie chart diagram is designed to help IT analysts determine the location and root cause of a failure based on graphical visualization and statistical probability. Figure 5 shows an example of an Invariant Analyzer Anomaly Pie Chart. The pie chart is divided into two parts. The inner pie chart identifies the tier of system infrastructure on which the failure is most likely to be occurring, such as a Web server. The outer ring shows which specific servers are experiencing large numbers of anomalies and are the most likely sources of the problems.

FIGURE 5

Invariant Analyzer Anomaly Pie Chart



Analysis target: Web3Tier

Item	Anomaly score
WEB...	289
WEB...	182
WEB...	120
WEB...	91
WEB...	80
WEB...	70
WEB...	63
WEB...	62
WEB...	58
WEB...	58
WEB...	54
WEB...	54
WEB...	48
WEB...	47
WEB...	39
WEB...	38
WEB...	31
WEB...	28
WEB...	19
WEB...	18
DB02	90
DB03	88
DB04	76
DB01	17
AP03	23
AP02	19
AP04	16
AP01	8

Source: NEC, 2011

Invariant Analyzer: Key Benefits

Invariant Analyzer is based on sophisticated analytic technology that discovers and analyzes relationships among monitored variables rather than relying on variable thresholds to generate alerts. Invariant Analyzer can provide a wide variety of key benefits, including:

- **Automated discovery.** Invariant Analyzer discovers relationships among variables without the need to set up and maintain thresholds, thus avoiding manual processes and helping achieve short implementation times.

- **Use of existing monitors.** Invariant Analyzer uses standard performance metrics data available from existing monitoring tools.

- **Improved problem determination.** Invariant Analyzer can improve problem determination and root cause identification by providing ranked performance behavior anomaly detection and encompassing a comprehensive view of systems performance through analysis of large volumes of performance metrics across multiple IT domains.

- **Predictive analysis.** Invariant Analyzer helps avoid slowdown and disruption by detecting conditions that could lead to critical performance problems in the near future (i.e., failures that are currently "silent" but may manifest themselves soon).

- **Ability to work with other products.** Invariant Analyzer integrates with other monitoring products that provide additional levels of detail such as CA APM (former Wily Introscope).

CHALLENGES AND OPPORTUNITIES

NEC has invested in sophisticated analytic technology with Invariant Analyzer. As stated earlier, automated analysis is becoming a critical requirement for ensuring the performance and availability of large, complex, dynamic IT infrastructures including virtualization and cloud. Invariant Analyzer is very much aligned with this trend and addresses the increasing need for automated software tools.

The challenge for NEC is to identify, position, and promote Invariant Analyzer in a highly competitive and increasingly consolidated marketplace where a number of major vendors are moving in the direction of integrated stacks and their own management platforms. In this environment, Invariant Analyzer needs to become recognized as a complementary value-add to existing enterprise management solutions.

Key opportunities for NEC are to establish close partnerships with leading independent software vendors and system vendors and show how Invariant Analyzer adds analysis capabilities to other forms of performance management and integrates with other tools. Indeed, one important integration point is Invariant Analyzer's ability to use a wide variety of monitoring data sources from existing tools, such as CA APM (former Wily Introscope), complementing existing deployments.

SUMMARY AND CONCLUSION

Invariant Analyzer addresses a key requirement in operational IT systems: how to anticipate, identify, prevent, and resolve performance problems in highly complex environments. The focus on automation — including automatic discovery of invariant relationships — resonates with the need of IT organizations to manage increasingly large-scale system deployments with limited staff. Invariant Analyzer can play a major role in optimizing performance in virtualized and cloud infrastructures.

NEC should continue to demonstrate business value by showing how Invariant Analyzer delivers benefits such as performance assurance, reduced downtime, and faster problem diagnosis and resolution — all factors that contribute to improved business service delivery, more competitive interactive applications, and reduced IT operational costs. Quantitative examples and user case studies can play a strong role in demonstrating the business value and ROI of Invariant Analyzer.

USE CASE EXAMPLE

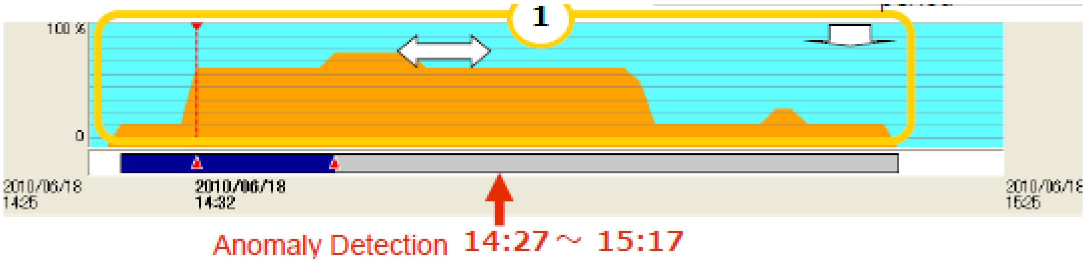
Diagnosing a Database Access Slowdown

NEC conducted a study with a major airline to show how Invariant Analyzer was used to discover the root cause of a major operational slowdown that was causing degradation to online accesses to a key database and terminating some accesses. The target system had 26 nodes, which were being monitored using a third-party solution. About 75 metrics per node — a total of 1,950 metrics — were being collected and analyzed.

According to NEC's analysis, Invariant Analyzer detected a strong anomaly occurring for a 50-minute period. Figure 6 shows the Anomaly Graph for this incident.

FIGURE 6

Access Slowdown Anomaly Graph



Source: NEC, 2011

Further analysis with Invariant Analyzer showed that the monitored metric associated with the primary WebLogic server had the highest ranking, indicating that this server was the source of the anomaly. Further investigation revealed that a database processing slowdown, due to engineers running a pre-process, was causing a WebLogic slowdown and message backlog. Indeed, WebLogic's pending messages increased rapidly and the message pending counter reached the upper limit. The anomaly ended after about 50 minutes. This corresponded to engineers completing a post-process, ending the database slowdown and allowing the WebLogic messages to be processed.

Copyright Notice