



***The eG Approach to
Root-Cause Analysis***

W h i t e P a p e r



Restricted Rights Legend

The information contained in this document is confidential and subject to change without notice. No part of this document may be reproduced or disclosed to others without the prior permission of eG Innovations Inc. eG Innovations, Inc. makes no warranty of any kind with regard to the software and documentation, including, but not limited to, the implied warranties of merchantability and fitness for a particular purpose.

Copyright

© Copyright 2003 eG Innovations. All rights reserved. eGurkha and eG ASPLite are trademarks of eG Innovations. All other trademarks, marked and not marked, are the property of their respective manufacturers. Specifications subject to change without notice



1. Introduction

One of the most important capabilities that an IT infrastructure performs for an administrator is root-cause analysis (RCA), i.e., the ability to be able to quickly pinpoint the root-cause of a problem from among a variety of symptoms. The term “correlation” is used to refer to the process of analyzing the symptoms of a problem to be able to determine its root-cause. Correlation techniques that can effectively perform root-cause analysis have been the subject of significant research by the network and systems management community in the recent past. This technical brief highlights the unique multi-phase correlation approach that eG’s suite of monitoring products incorporate to enable root-cause analysis in complex Internet/Intranet environments.

2. Defining Correlation

Most monitoring solutions purport to have capabilities for “correlation” and “root-cause analysis”. Many a times, when a monitoring solution supports root-cause analysis capability, it just means that the solution throws up a number of pretty graphs and charts that an administrator has to analyze and manually identify the root-cause of the problem. Sometimes, more than a half dozen analysis steps are required before an administrator can deduce the cause of a problem! Such manual correlation can only be performed by expert administrators and is very often based on the administrator’s intuition and experience.

To be truly effective, monitoring solutions must be capable of minimizing the workload on the administrator, by performing correlation as far as possible automatically. This would allow even a relatively inexperienced administrator to quickly identify and fix the root-cause of problems. The importance of root-cause analysis is highlighted by a Gartner study from two years ago, which determined that administrators spent over 80% of time to just identify the root-cause of problems! This study was just limited to problems at the network level. Considering the growing complexity of applications and end-user services, determining the root-cause of a service/application related problem probably accounts for over 99% of the mean time to repair a problem!

Many vendors use the term “correlation” to mean different things. Very often, correlation may just mean simple alarm suppression (e.g., if the monitoring system has detected a problem and a second alarm indicating the same problem is received, the monitoring system will suppress the duplicate alarm). Sometimes, alarm filtering is referred to as correlation – e.g., ignore alarms generated during maintenance times of a device or server. True correlation and root-cause analysis is based on the notion of “dependencies”. No element (network device, server, application) of an IT infrastructure operates in isolation. There are various forms of dependencies between network elements, between applications, between network elements and applications, all of which are used to support the end-user service. A true correlation engine that performs root-cause analysis must consider the inter-dependencies that exist in the target IT infrastructure and use this knowledge to identify the root-cause of problems.

3. Existing Approaches

3.1 Circuit-based Correlation

Many existing monitoring solutions have used the circuit-based approach to correlation. Applied mainly in the context of event-storms in network-centric environments, this approach involves considering all possible combinations of events that can be generated within a time window and building specific combination rules for each combination of events. HP OpenView’s event correlation engine is a classic example of a solution based on this approach. The main drawback of this approach is that in a real-world environment, it is impossible to predict all the possible combinations of events that can occur. Even if such a prediction were possible, constructing the correlation logic is usually a very laborious manual process, taking months of customization specific to the target environment. Even worse, even minor changes in the target environment could mean that significant recoding of the correlation rules may be necessary.



3.2 Symptom-Cause Correlation

A second set of correlation solutions are based on having operators aid the monitoring solution by mapping symptoms to root-causes. Based on the past history of symptom to root-cause mappings (often referred to as a “failure signature”), these solutions are able to recommend root-causes for problems. The mapping of failure to root-causes can be done either at application development time or during the operation of the system. Neural net classifiers, case-based reasoning, and simple rule-based analysis are some of the techniques that have been used to map a current problem to the closest known failure signature. The main limitation of this approach is that the learning period for determining all possible failure signatures can be significantly long. Secondly, failure signatures are often defined in a time-dependent manner (e.g., all the failures occurring within a specific time window). In reality, Internet/Intranet environments tend to be very dynamic and there may time differences between the occurrence of a problem and its propagation to other places of the network. In such cases, if the correlation approach is not robust enough to handle the occurrence of problems at different times, this could lead to incorrect problem diagnosis.

3.3 Domain Specific Correlation

A third form of restricted correlation relies on specific domain knowledge. An example of this correlation technique is most often found in network monitoring tools. When a single router fails, all other elements behind the router may be reported as being unreachable. The monitoring solution uses a network topology map (say built using the traceroute utility) to identify the specific router that may be the root-cause of all the failures. However, this technique is limited to addressing network-level problems alone. In fact, very few solutions offer true correlation across the network, server, and application layers.

4. eG’s correlation approach

In designing the correlation techniques embedded in eG’s monitoring solution, the following goals were considered:

- The correlation approach should involve minimal configuration and customization, so that the effort involved in setting up the correlation logic is minimal;
- The correlation approach must be capable of identifying the root-cause of problems and perform automatic infrastructure triage relating to the network, system, and application layers (not just the network);
- The correlation must not be restricted to “silos” based on the organizational structure; it is critical for the correlation approach to not be constrained by the organization of the network maintenance team;
- Automatic infrastructure triage with little or no human intervention should be possible;

The distinguishing feature of eG’s automatic infrastructure triage and correlation approach is its simplicity and elegance. By codifying common logic that operators are most likely to employ in order to diagnose a problem, eG’s correlation engine performs correlation and root-cause analysis.

There are three key elements of eG’s correlation approach:

➤ **A layer model representation of any network device or application:**

The often-stated Open Systems Integration (OSI) model is a great way of representing the inter-dependencies between the different layers involved in supporting a network or application function. According to this model, the layer above depends on the layers below for its proper operation. Therefore, there is a strict hierarchy between the layers that comprise of an application. However, the OSI model itself is idealistic. In reality, the OSI model has to be adapted for different network devices and applications. Moreover, the OSI model is mostly focused on the protocol stack and does not take the host on which the application executes into account.

The eG suite represents each network device or application as a layer model – wherein the layers represent components that are involved in supporting the network device or application’s functions. Since one application (or network device) is distinct from another (e.g., an Apache web server is different from an Oracle database server), the layer model is uniquely defined for each application (or network device).



Figure 1: eG's layer model representation

To understand the utility of the layer models for correlation, consider the example in Figure 1. This figure shows the layer model of a web server. eG agents report measurements regarding the web server. These measurements are mapped to different layers of the web server model. Based on the measurements, Figure 1 indicates that the WEB_SERVER layer is experiencing a problem and so is the NETWORK layer. The correlation logic applied to the web server layer model indicates that the root-cause of the problem is probably at the NETWORK layer. The eG manager applies this correlation logic and performs automatic infrastructure triage. The alarm window (shown in Figure 2), which presents the results of the correlation by the eG manager, indicates that the NETWORK layer alarm is the root-cause of the problem and is hence marked as a CRITICAL alarm. The WEB_SERVER problem may be caused by the NETWORK problem and hence, the web server alarm is marked as being of a lower priority.



Figure 2: Alarms indicating that the NETWORK layer problem is of a higher priority than the WEB_SERVER problem

The eG suite has pre-built layer models for most popular applications. The advantage of this approach versus the circuits-based correlation approach described earlier is that the layer model defines a template of events and the correlation logic. This template is just re-applied to every instance of a particular application. The circuit-based approach may involve constructing the same correlation logic again and again for each instance of an application.

➤ **Service topology representation of inter-application and application/network dependencies:**

The second part of eG's automatic infrastructure triage and correlation approach relies on the notion of service topologies. While the layer model represents dependencies between the layers of a single network device or application, a service topology represents the dependencies of one application on another, or between an application and a network device. A service topology is a graph that defines the data flow from the user to the different applications and network elements that are involved in supporting the service. The topology is logical in the sense that even applications that are executing on the same host are considered distinctly (because they are independent entities that can each function or fail).

The basic premise behind the use of service topologies is that applications and network elements rarely function in isolation in an IT infrastructure. A failure in one application or network element can ripple and affect all the other applications or network elements involved in supporting the service. Hence, it is important to understand the interdependencies between the applications and network elements involved in supporting an end-user service.

The eG correlation engine uses the service topology representation to identify which among a group of applications that are involved in supporting a service could be faulty. The application groups involved in supporting the service may even span multiple domains. This correlation approach is best explained using an illustrative example. Figure 3 shows an end-user service – in this case a web site – that is having a problem.



Figure 3: A problem with a web site named buy.abc.com

Further investigation reveals that users are unable to login and add to cart via the web site and are experiencing problems (see Figure 4). To understand the cause of this problem, click on the problematic transaction.



Figure 4: The list of transactions associated with the web site buy.abc.com

The topology graph of the web site (see Figure 5) depicts that this site is supported by a web server, using an Oracle database backend. In the example of Figure 5, several applications are experiencing problems. The arrow between these applications indicates the direction in which problems tend to propagate. In this case, since the web server is using the Oracle database, a slowdown/failure of the database can impact the all the other related applications. Hence, to further troubleshoot the problem, click on the Oracle database server.

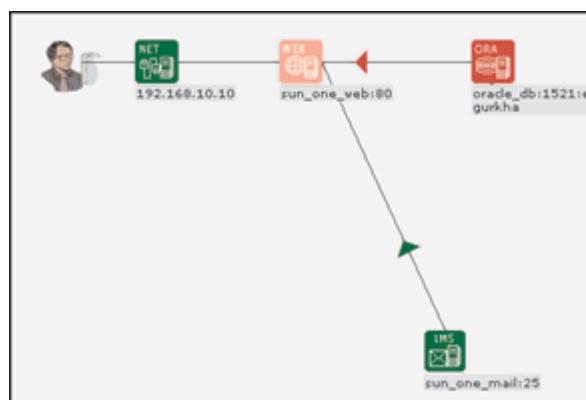


Figure 5: Topology graph

Doing so reveals the layer model for the Oracle database and the exact layer that is experiencing the problem (see Figure 6).

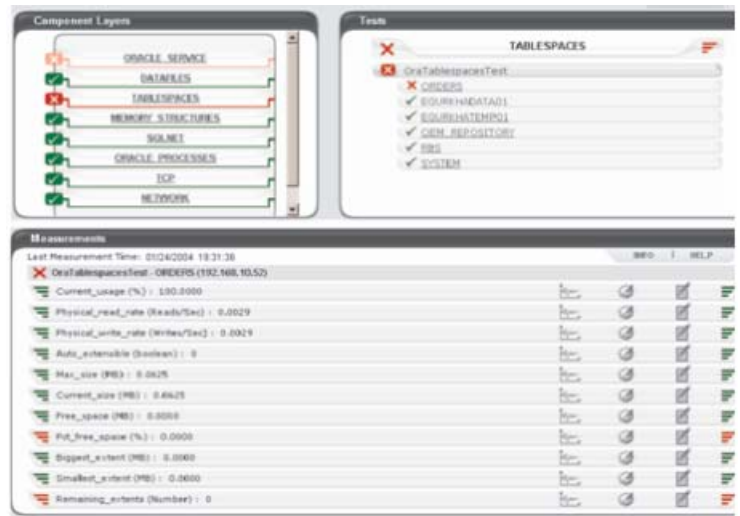


Figure 6: The layer model of an Oracle database server indicating the problematic layer

Notice in Figure 7 that the eG manager has automatically performed the steps mentioned above and has correctly tagged the Oracle server alarm as the CRITICAL problem. The alarm associated with the web transactions layer is marked as a low priority alarm, since it is not viewed as the root-cause of the problem.

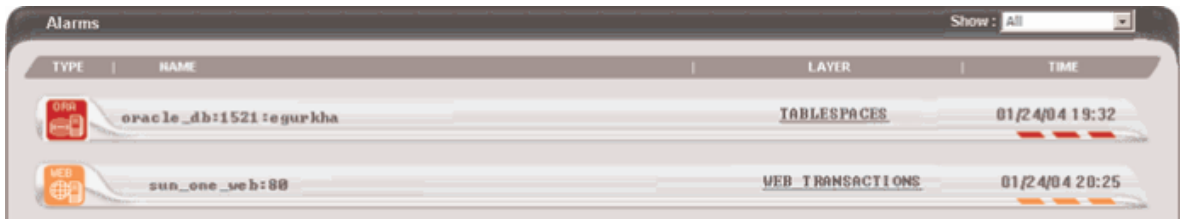


Figure 7: The list of alarms pertaining to the site buy.abc.com

To further minimize the alarms reported to the user, the eG manager implements various forms of alarm compaction. First, multiple alarms pertaining to the same layer are compressed into a single alarm. Moreover, if the same layer is viewed to be affecting multiple applications, the root-cause of the problem is reported only once.

➤ **Router topology representation to handle network dependencies:**

A service topology represents the logical (software-level) interdependencies between applications and between applications and network elements. The performance of an end-user service may also be impacted by the network elements that are involved in supporting the service. To handle network dependencies, the eG manager discovers and tracks network-level connectivity. Based on the discovered or manually configured network connectivity and routing information (referred to as the router topology), the eG manager performs a third phase of correlation from where it can prioritize between external network and application alarms (e.g., if a server is not reachable, and a router along the path to the server is not reachable, then based on the network topology, the router connectivity is marked as the CRITICAL problem).



5. Key Benefits of eG's Correlation Approach

- A significant reduction in problem durations, resulting in better service quality and satisfied users;
- eG's patented automatic infrastructure triage capability makes problem diagnosis very simple. Considering inter-application and network interdependencies, eG is able to automatically differentiate between the cause and effect of problems, so service operators can focus on the cause, rather than being distracted by the effects;
- Taking a holistic approach, the eG suite performs integrated multi-tier monitoring allowing IT operators to track the quality of the business services and relate this to critical network, system, and application performance. Pre-defined models for over fifty popular software applications ensure that service operators can start managing their infrastructure, without needing to acquire a great deal of expertise;
- The eG suite allows IT operators to track the quality of the business services and relate this to critical network, system, and application performance. Pre-defined models for over fifty popular software applications ensure that service operators can start managing their infrastructure, without needing to acquire a great deal of expertise.
- eG's layer model representation for different applications and network devices enables clear demarcation of the problem, with minimal effort. This capability eliminates finger-pointing across domains;

6. Conclusion

Effective correlation and accurate root-cause diagnosis has been a key challenge for network and systems management solutions for many years. The challenges in root-cause diagnosis have only grown over the years, with the growing complexity of the software applications and services that operate on top of the Internet and Intranets. The organization of network maintenance teams into vertical silos, with one set of operators being responsible for the network, another set being responsible for the servers, a third set taking care of the databases, etc., has further compounded the problems. Yet at the same time, customer expectations of service quality and guaranteed performance have been ever increasing. eG's suite of performance monitoring solutions adopts a unique, multi-phase approach to correlation and root-cause diagnosis. The simplicity, elegance, and adaptability of this approach to different networking environments differentiates the eG approach to correlation and root-cause diagnosis.