




Proactive, End-to-End Performance Management
Finding Issues Before Your Users Do

Indicative Software, Inc. 
724 Whalers Way, Building I
Fort Collins, CO 80525
970.530.0790

info@indicative.com www.indicative.com

➤ Executive Summary

Troubleshooting within the enterprise IT environment is enormously challenging. The infrastructure becomes more complex as IT adds new capabilities such as J2EE, Citrix and .NET applications potentially front ending a mainframe and deploying across geographies. When problems occur, infrastructure monitoring tools can generate a blizzard of event reports that can literally bury the IT staff's ability to pinpoint the source, slowing troubleshooting response to what is happening in the environment. A variety of alternatives exist for improving the way IT manages the event storm. These range from multi-domain bridge calls, to implementing simple event filters, to developing a comprehensive event correlation system. Each of these alternatives has benefits and drawbacks. However, any approach that keeps the IT staff focused on infrastructure events will inevitably be resource-intensive and may not result in the best response to user problems, let alone preventing future problems.

A better approach would start with an end-to-end service delivery view modeled to the underlying infrastructure and include proactive testing from the users' perspective. A hierarchy of infrastructure components is constructed such that inter-dependencies are clearly visible to IT administrators. Troubleshooting is in terms of the impact at the highest levels of the hierarchy, effectively prioritizing infrastructure events according to the relevance for the user perception of the service or end-result. Such an approach creates a user-centric view of the infrastructure that by definition includes the relevant components and measurements underlying the user experience. Some management systems purport to deliver this capability, but IT managers should look closely and question vendors to avoid potential monitoring blind spots or costly and inflexible deployments.

Table of Contents

Executive Summary	02.
Challenges in the IT Environment	03.
Troubleshooting Alternatives	04.
Recommended Approach:	09.
Proactive, End-to-End Management	10.
Summary	14.
Planning Guide	15.



Indicative Software, Inc.
724 Whalers Way, Building I
Fort Collins, CO 80525
970.530.0790

→ Challenges in the IT Environment

Complexity and event volume

Making sense of changes and events within an increasingly complex IT infrastructure is one of the primary challenges facing IT organizations. Users view applications and services in simple terms. Is it available? Is the performance or response time adequate or am I losing productivity by waiting for the system? But many separate components can contribute to the user experience of each application or service.

For example, finance administrators may use an order entry application in regional offices of a global sales organization. The users access the application via a Web browser traversing security perimeters and the corporate backbone to the central order processing server farm on a LAN in another country and a mainframe in the corporate data center. Network, system, application, mainframe, database, and desktop services in several locations need to operate within acceptable limits for the user to receive satisfactory performance levels.

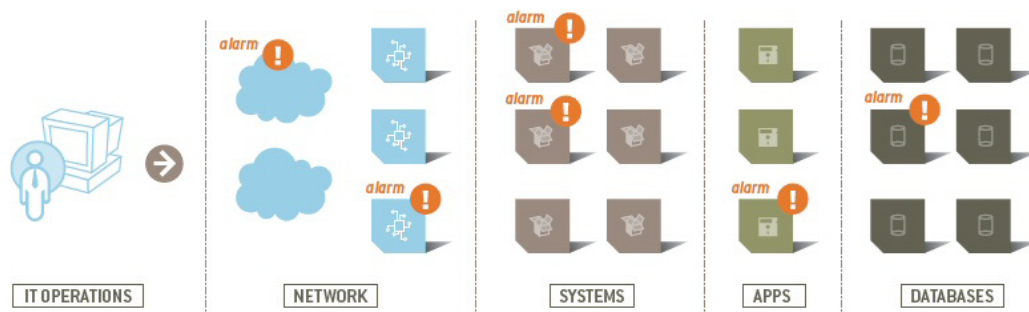
When a problem occurs, IT must sort through the layers of the infrastructure to identify, prioritize, and fix it. They may have tools like HP's Network Node Manager for network management, BMC Patrol or Microsoft Operations Manager for systems monitoring, Quest for database monitoring, etc. When any element exceeds its pre-set alarm threshold, the respective monitor generates an event that appears on the management console for that part of the infrastructure. As a result the IT staff gets timely data on tiny changes in the environment. In reality the sheer number of events can be overwhelming since a single alarm can propagate through the environment and generate numerous additional alarms. In addition, sometimes the root cause of an event is the cumulative effect of several smaller changes in components that approached but did not exceed their own respective threshold. Such problems can be difficult to trace, or they can be intermittent and difficult for IT to replicate.

The complexity can escalate yet again whenever IT adds a new application or modifies an existing one. Or they may Web-enable existing applications that typically must interact with components outside the corporate firewall. This further reduces visibility in a troubleshooting situation.



Lack of correlation across technologies

For some organizations the current challenge is to extend monitoring to areas of the infrastructure where they currently have blind spots. For others, adequate monitoring may already be in place. But the monitoring tools typically focus on particular technology domains. Combined with their high degree of granularity, the tools provide IT with plenty of data but not enough insight to understand the big picture relevance. IT administrators must review the incredibly detailed data on a series of management consoles, and then use their experience to correlate the data to build a coherent picture of what is happening across the environment. Frequently when tough problems arise, management initiates bridge calls between domain owners leading to hours or days combing through event-centric data.



➔ FIGURE 1: IT infrastructure is increasingly complex and interdependent.

This is not to question the value of monitoring tools. But once a comprehensive monitoring capability exists, users will still find problems that the tools miss. Simply adding more tools to aggregate the monitoring data will not help IT be proactive and improve the troubleshooting job. IT needs something that takes the users' perspectives and helps it make sense of the information it gets from the tools it already has.

Faced with this complexity and the daily task of sifting through the hundreds or many thousands of infrastructure events, IT organizations today are considering their alternatives. They need to improve the way they interpret events, troubleshoot, and resolve service complaints from users. A one-size-fits-all solution does not exist despite vendor marketing claims. Depending on their resources and management process maturity, organizations may have already begun to implement and benefit from one or more of the approaches discussed in the following sections.

→ Troubleshooting Alternatives

Reactive practices

Adding more people

The first way to filter infrastructure events is simply to hire more people for your troubleshooting team and partake in bridge call meetings as appropriate. If IT can afford it, this option is perhaps the easiest to implement and has the advantage of minimal change to the existing environment. In fact, IT organizations already rely on the human expertise of their people to interpret and correlate events. Adding people to handle the increased workload is a logical next step. Having more people focused on particular technologies means you deepen the understanding of how those technologies behave and what conditions to look for when user problems occur. You can also dedicate people to a meta-role looking across technology silos to understand the broader implications of localized events.

73% of problems reported by end users through the service desk are not detected by infrastructure management tools

Forrester Research, April 2004

Yet in today's world, this option is not cost effective for many organizations. In addition to affordability, IT managers need to consider the point beyond which adding more people does not achieve a corresponding improvement in problem solving. (See *Mythical Man-Month* by Frederick P. Brooks, Jr.) The human expertise in the organization will always be its most vital asset. However, proactive troubleshooting processes will really improve only if the IT team can capture and institutionalize the expertise of its best people. Cross-training, sharing best practices, and other communications processes become more important for mining the team's collective knowledge base. Ideally the team should find a way to build this expertise into its management tools as well as expect this knowledge from its management vendors.

Filtering through a trouble-ticket system

A trouble-ticketing system can act as a rudimentary event filter by capturing some of the human expertise in the organization. Monitoring tools are typically configured so they generate trouble-tickets for every problem. The trouble-ticket filtration system can be set up to prioritize specific classes of problems, or problems occurring in relation to pre-scheduled IT tasks, and to remove duplicate tickets. This in itself can reduce the randomness of the IT event storm. However, hundreds of tickets may still be generated for a single problem. Most of these represent symptoms, for example, a specific element exceeding a threshold condition, but the ticketing system doesn't point to the underlying issue that caused the threshold to be exceeded.

The objective of any automated filtration system is to reduce the volume of events that IT staff must interpret. Such systems can provide



Indicative Software, Inc.
724 Whalers Way, Building I
Fort Collins, CO 80525
970.530.0790

rudimentary classification, but inherently they treat all tickets equally without knowing the relationship between them. Creating those relationships - a sub-ticket to “super-ticket” view - can be done programmatically but it requires enormous skill in defining the relationship rules. And the rules are valid only if the environment hasn't changed since their most recent update.

On the positive side, ticketing systems can be linked to a knowledge base that catalogs recurring symptoms and underlying problems. This gives IT a reference tool for finding likely root causes and resolving problems based on previous history.

The objective of any automated filtration system is to reduce the volume of events that IT staff must interpret. Such systems can provide rudimentary classification, but inherently they treat all tickets equally

Event correlation

Event correlation engines are robust filtering systems that are based on comprehensive and sophisticated dependency rules than simple trouble-ticketing filters. However, they attempt to infer rules and higher level behavior from a bottoms-up perspective on infrastructure events. Plus, it is very difficult to encapsulate the knowledge of experts into rules especially as the infrastructure changes on a daily basis.

Nonetheless, some organizations have achieved substantial reductions - even as much as 90 percent – in event volumes through event correlators. But implementing a system that can generate such results is not easy. Dependency rules can be difficult to articulate, since the human expertise behind them also includes judgments based on experience. Building a correlation engine is time and resource-intensive. Once built, these systems can also require a dedicated maintenance team familiar with the proprietary programming language used for encoding the rules. Refining the rules or adding new ones can become a massive task. It is not uncommon for an enterprise to require a dedicated team to build and maintain its correlation rule set.

The maintenance task cannot be over-emphasized. It points to one of the biggest challenges for event correlation systems. Most IT environments are quite dynamic, with upgrades, new equipment, applications or services coming online daily. The correlation system needs to be able to update dependency rules as needed. Other than changes to the infrastructure, updates may also be driven from IT's clearer understanding of specific dependencies. In that case they may be able to significantly enhance the effectiveness of the correlation system by modifying the relevant rules.



Indicative Software, Inc.
724 Whalers Way, Building I
Fort Collins, CO 80525
970.530.0790

Of course some environments are more stable and would have less need to update dependency rules. For example, financial services providers may decide the priority for system availability outweighs the benefits of periodic downtime to refine the correlation system. Also, IT managers might favor the known limitations of the current system over estimated but unproven benefits of a modified one.

Although IT expertise is the raw material for building the correlation system, it may also get trapped inside the system. In a sense, event correlation systems can become black boxes that crunch a large number of events and alarms and filter a smaller amount through to the other end. The rules used by the correlator are hidden from the IT staff so it is difficult to apply additional human expertise. This can make it more difficult to troubleshoot intermittent or first-time problems, because the way the correlator arrived at the filtered set of events is not visible. In the past, a by-product of the problem solving process no doubt included a fair amount of knowledge sharing and a resulting increase in staff expertise. Some of this knowledge sharing may be lost when much of the expertise and rationale for decisions is hidden.

Also, correlation engines inevitably lag technology innovations. Because of the difficulty in updating rule definitions, the engines will contain rules for the oldest and best understood infrastructure components, but perhaps not the latest web services or applications. Murphy's Law suggests the majority of problems would arise from those components that are least defined in the correlation engine.

Again, even if you have the resources to create and maintain a robust correlation engine, it will be operating on low-level events. Meanwhile your staff needs help tracing the events to their impact on users. Even if the IT staff is exposed to a somewhat milder event storm, the correlation engine still can't help them interpret the higher-order impact of what they are seeing. By analogy, suppose the information system being monitored were a human body with a weak heart. But the IT organization gets data only from molecular or cellular activity, where millions of cells are created and destroyed each minute. Even if the number of cellular events can be filtered by 95%, IT still must deal with millions of events with little visibility into their effect on the heart condition.

Event consoles and service dashboards

A class of tools has become available that sit on top of management systems and event correlation engines to provide a form of business-level event interpretation. These tools give an overall availability status for a



well-defined IT service. They are sometimes called Business Service Management systems (BSM). Ideally they take the reduced event stream from the individual tools, apply additional dependency rules at a more macroscopic level, and give IT guidance on business impact and troubleshooting directions.

Industry analysts observe that less than 10 percent of IT organizations are able to successfully implement BSM systems. The biggest prerequisite for IT is to have already defined the user level services to manage, ideally through end-to-end service level agreements with the business entities that use IT services. IT also needs to use event correlators to reduce events to the most critical for the performance of the service in question. BSM systems typically don't include auto-discovery capabilities, so the process of selecting and loading infrastructure objects into service-level views is incredibly complex. IT organizations with well-established procedures and naming conventions will have shorter BSM implementation schedules.

Organizations without such well-established processes or without event correlation systems must go through a longer term implementation effort starting with re-engineering their own processes. First they must define service-level expectations through detailed discussions with their business unit users. They would need to inventory the entire infrastructure from the bottom-up and build object models. Then they need to create the interdependency rules for the objects. The rules are then inferred into higher level performance models for the services they want to manage. Given the length of the process, IT also must hope that the environment or service expectations have not changed drastically during the implementation phase.

Once the implementation is complete, the BSM system is still based on hard-coded rules that take a bottoms-up perspective on how infrastructure affects a given service. The rules are hidden and difficult to modify. BSM systems are best seen as toolkits that require their own configuration and maintenance investments. Determining the magnitude of the BSM configuration task is a useful exercise when evaluating BSM solutions.

In addition, some of the available BSM toolkits have evolved from the specific technology expertise of the vendors. This can result in a BSM system that gives availability information for focused parts of the infrastructure such as networks or systems, but may be blind to problems in other areas.



Indicative Software, Inc.
724 Whalers Way, Building I
Fort Collins, CO 80525
970.530.0790

The key success factor for vendors is less about who has the most monitors and more about who can build real-time links between the IT assets and the business processes, really defining what the user is experiencing.

**Technology: Infrastructure
Software, Systems
Management Handbook
Goldman Sachs, 2004**

The importance of proactive testing and context

The key drawback in the approaches discussed above is they tend to be both reactive as well as lack context given the dependence on infrastructure events. This is the result of passive data collection coupled with a bottoms-up, or inside-out perspective that collects information from the lowest levels of the IT infrastructure and attempts to make sense of it. Even event correlators or availability consoles that attempt to build a higher-order model use these lower level building blocks as a starting point. This is a labor intensive undertaking. What's more, the result can be a black box that reduces visibility instead of shedding light on fruitful paths for troubleshooting.

So far, IT management systems have attempted to filter, aggregate, or somehow interpolate infrastructure events into a model that represents the user experience. A better approach would be to start by defining the desired end result, in terms of application performance or use of a service, constructing a model or view of the relevant underlying portion of the IT infrastructure and continuously exercising it from that user perspective.

Recommended Approach: Proactive, End-to-End Performance Management

Defining performance in terms of user experience

A more effective approach to managing infrastructure events starts with a definition of a service or application from the perspective of the user. The application or service can be defined in whatever terms are most relevant for the user, and at whatever level of abstraction makes sense for IT to manage - from technology to higher levels like order processing performance. The service definition then becomes the starting point for constructing a logical view of the infrastructure underlying service delivery.

Such a view consolidates the infrastructure into subsets around specific transactions, applications and functionality. The view is hierarchical, and the relevance of lower level events becomes apparent based on their impact at higher levels especially when coupled with active tests. Once the hierarchical model is developed, IT can troubleshoot events by testing the logical transactions of the user community.



Indicative Software, Inc.
724 Whalers Way, Building I
Fort Collins, CO 80525
970.530.0790

We can call this approach proactive, end-to-end performance management. End-to-end means it gives IT the ability to look across technologies and platforms and across existing management tools in a single console and find issues before their users do. End-to-end performance management provides a real-time measurement of relevant user experience by explicitly testing while reporting on the underlying infrastructure and inter-dependencies health. Relationships can be seen and changes in the environment are immediately apparent in the single console.

A hierarchy of application and service dependencies

Using the example of an order entry service, the top level of the hierarchy is the service itself. At the next level are the common transaction paths that comprise the service and the servers that implement the service. Each of these servers in turn is composed of the server software, OS, and network interfaces which support the order entry implementation. Finally, dependent elements of the infrastructure (e.g. DNS and authentication servers, backbone throughput, database performance and availability) are also tied into the service model. At the leaf nodes of this service tree are the actual measurements which monitor fundamental aspects of the health and performance of the transactions and each of the components represented higher up in the tree.

In the end-to-end performance management model dependencies, relationships, and actual measurement data is brought together to form a complete picture of what makes up the service. The model expresses the relationships among the various elements of the infrastructure and the services they support. It also defines the measurements that are actually relevant to the user experience of the service. Among other benefits, this eliminates collection and storage of data that has minimal relevance or usefulness. For example, the interface portion of the Internet Engineering Task Force MIB II contains a lot of data, but only a small amount of that data is actually relevant to determining whether interface traffic is impacting the actual quality of any given service.

Proactive transaction tests

To truly understand user experience, the system proactively tests common user transactions as well as collects actual experience data. Proactive or synthetic transactions are generated by a GUI-driven record and playback tool that is directly integrated with the model to optimize correlation between user experience and infrastructure health. Transaction testing should be broken down into individual steps and traverse the service delivery chain, inside and outside the enterprise including third-party providers. Actual transaction capture provides a robust troubleshooting tool for application developers and operations.



Managing performance instead of interpreting events

Measurements defined within the end-to-end performance model represent factors that determine the user experience for one or more services. The IT staff can establish statistical baselines by time of day and day of week. They can set multi-tiered thresholds for these measurements. When these thresholds are exceeded, alerts are generated, notifying the administrator as to what service is impacted and what aspect of the service is out of specification before it affects users.

The end-to-end performance management method also exercises the infrastructure components and interdependencies to test actual behavior relative to expected behavior. If the performance is outside the boundaries at the high level, the administrator can drill down and evaluate performance characteristics at the next lower level. From there troubleshooting continues down the chain until the root cause is uncovered. A fundamental benefit comes from the proactively exercising the service and underlying components including interdependencies to uncover problems before they are noticed by users.

Rapid troubleshooting across technology domains

The end-to-end approach is tops-down, measuring higher level performance and from there looking lower within the hierarchy for underlying causes. In a sense the end-to-end management model is a black box of the user's perspective of the service. But rather than hiding the inner workings of the box, end-to-end management enables troubleshooting by stepping into the next layer of the box and analyzing performance from there. This amounts to stepping through a specific series of nested boxes at successive levels of the hierarchy. Alarms occurring in the infrastructure are viewed in the context of the current level of the hierarchy, or current box. Out-of-boundary conditions at each level guide the next step in troubleshooting based on the pre-defined infrastructure components whose behavior affects the current view.

In the order entry example used earlier, if the response time is below normal, the end-to-end management model guides the operator directly to the measurements that indicated the problem. Consider the following scenario:

1. An operator is alerted to an availability problem in order entry. The model expands automatically to show what test detected the event, based on a threshold being exceeded.



Indicative Software, Inc.
724 Whalers Way, Building I
Fort Collins, CO 80525
970.530.0790

2. The operator examines the model to find that only one server is experiencing a problem. This immediately eliminates many aspects of the infrastructure that might have contributed to an order entry performance problem. For example, backbone performance, authentication server problems, and content availability would have impacted all order entry servers equally.
3. Further examination shows that this server in particular is experiencing greater than acceptable response times. This increase in response time has caused the response tests to time out. This represents a common problem – identification of gray faults. In this case, users perceive that order entry is unavailable. This is due to client timeouts caused by poor server response times. However, tests specifically designed to test for order entry outages fail to indicate a problem – the order entry application is actually available, just very slow.
4. By examining the individual components of the order entry server response tests, it can be seen that server response time is the primary contributor to the overall increase in response time. This indicated that the server in question is having problems servicing requests. While the TCP/internet protocol (TCP/IP) stack is able to accept connections normally, as indicated by nominal TCP connect time values, the server is not responding in a timely manner. This further eliminates local network, interface, and some OS configurations (such as allowed number of TCP connections) as a potential cause. This leaves the server itself or resources required to service the request as the only remaining sources of the problem
5. No other thresholds triggered events to lead to the root cause of the problem. However, a quick look at the system component of the model reveals that free memory is very low, probably too low to fork another server process to handle the order entry request. Further analysis, including possibly adding memory to the server machine, will be required to resolve the problem.
6. At the end of the diagnostic process, it became clear that a threshold on free memory would help identify this problem in the future. A threshold can easily be set that will trigger an event if free memory on any order entry server goes below a given percentage.



The previous scenario illustrates how the end-to-end management model facilitates performance testing and root-cause diagnosis. Using an end-to-end management model can help refine the measurement and threshold configuration to provide continuous improvements in problem detection and resolution. It also shows the items in the model that lead to the root cause and finds issues before the users do.

Visual correlation

The logical, hierarchical end-to-end model can show the group status at any level of the dependency tree. In the previous example, IT operators had an indication of the health of all servers even though one server experienced greater than acceptable response times. The model can also be expanded to show all users who may be affected by a given infrastructure event. Likewise, server tier performance can be aggregated to determine whether individual health is or is not relevant. Such cross-correlation can help IT staff prioritize their efforts according to which user problem has the biggest impact on the business.

Template driven set-up and automated discovery

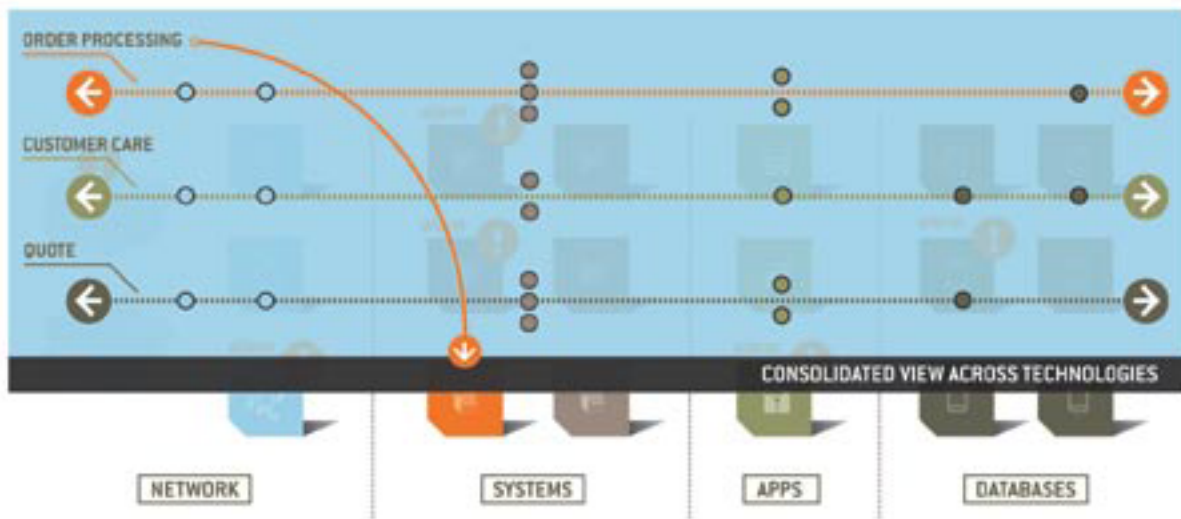
The user and service perspective starting point makes end-to-end management systems easier to set up than an event correlation engine. The IT staff can use templates that are populated by objects defining the relationships between infrastructure elements at each level of the hierarchy. Transaction testing and recording playback is GUI-driven and handles most application environments including Web, J2EE, .NET and Citrix environments. The result is a visual correlation capability that captures the IT expertise and makes it highly visible to IT staff using the system. End-to-end management is also easier to modify because unlike event correlation engines the dependency relationships can be changed by replacing a template rather than rewriting and retesting scripts. A template driven system is easier to configure out-of-the-box and less expensive to maintain.

An auto-discovery facility makes the end-to-end management more adaptable to changes in the IT environment. It can update itself automatically as new components are added to the infrastructure, and reconfigure dependencies accordingly.



➔ Summary

End-to-end management is not just another monitoring tool but instead a means for improving user performance while leveraging your existing investments. End-to-end management makes visible all the relevant dependencies among components that support a given IT service and proactively tests from the users' perspectives. Creating such a view and active data collection mechanism can be streamlined through the use of templates that quickly construct the service hierarchy. Relationships can be easily seen across levels, and forward and backward across an event chain for quicker problem resolution. This approach starts from the users' perspective. By defining the relevant parts of the infrastructure for each service, an end-to-end system helps IT organizations avoid getting lost in a storm of often irrelevant events.



➔ FIGURE 2: End-to-end quickly pinpoints the relevant issues



Indicative Software, Inc.
724 Whalers Way, Building I
Fort Collins, CO 80525
970.530.0790

➔ Planning Guide

The following questions may be useful in choosing between alternative ways to manage the increasing volume of IT infrastructure events and improving end-user performance.

IT self-assessment

1. Do users find issues that your management tools miss? Do you explicitly test end-user experience and model that to the underlying infrastructure?
2. How dynamic is the environment? How often do you add applications or new systems? What is the trade-off between hard-coding an event correlation system to streamline current troubleshooting, and the need to update it later?
3. What is the degree of IT process maturity? Do you have well-established processes that can be captured in end-to-end management templates?
4. What is the degree of service availability problems? That is, are the core issues around availability or more around reaction time and streamlining troubleshooting?
5. If availability is the main concern, are there blind spots in the current infrastructure monitoring toolset?
6. Do you have the ability to add more IT staff to handle the event filtering load?
7. Have you evaluated the risk of the expertise contained in your best people and established processes for sharing it more broadly in the organization?



→ **Questions for IT management software vendors**

1. What is the installation and implementation length associated with the vendors proposed solution? Is it more than a few weeks?
2. For an event correlation system, what is the likely scripting the IT staff will still need to do after the correlation engine has been implemented? What level of training and effort is necessary to make changes?
3. What is involved in modifying the vendor's system due to better understanding of dependencies, or to a change in the infrastructure such as a new application? Are these dependencies hidden or visible?
4. Does the vendor claim to have an “end-to-end” solution but is not able to include data from all dimensions of your environment? Are there still gaps?
5. Does the vendor monitor actual and synthetic transactions of your application environment? Is the transaction tool integrated with the end-to-end model or is that another console?
6. Is the vendor offering a “Manager of Managers” Console? This is often just a higher-level shell on the same uncorrelated, bottoms-up view of the infrastructure. Will users still find issues that the tools miss?
7. What is the origin of the vendor's solution? Did they evolve from a network-centric or a systems-centric view of the environment? Can they provide you the insight and cross-technology correlation you need to make sense of the data you get from the tools you already have?

