NetworkCare
The knowledge behind the network

Lucent Technologies
Bell Labs Innovations

The VIA Architecture: A Fresh Perspective on Application Performance Management

A VitalSoft Whitepaper

The Application Management Challenge

The role of business applications in today's corporate environment has never been more important—or more visible. From e-mail to e-commerce, more and more businesses are employing networked applications to support their day-to-day operations. As a result, managing and monitoring the performance and availability of these applications has become a critical part of IT's charter.

As IT is quickly discovering, however, these applications are highly decentralized, and the distributed and complex nature of the architectural components makes most traditional application management solutions obsolete. While a number of solutions exist to address application performance in older mainframe-based, host-centric environments, these techniques do not address the management challenges of today's predominantly distributed application environments. In addition, recent developments within the network infrastructure — technologies such as switching, virtual private networks (VPNs) and remote access — have hampered traditional management solutions. Throughout the corporate world, IT departments are searching for a consistent set of tools and methodologies for monitoring and managing the applications that run over these new, complex networks.

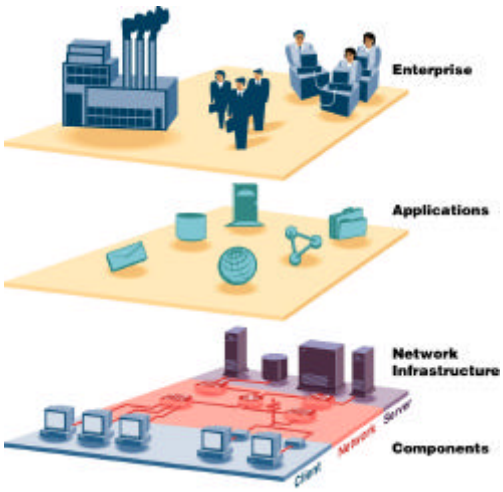

Figure 1. The Distributed Networking Environment

This search, however, is complicated by a number of obstacles — issues that need to be overcome before a true distributed application management solution can become a reality. These challenges include:

- **Complexity:** The complexity of today's applications, combined with the number of people using them, continues to grow at an alarming rate, with no end in sight.
- **Trend Toward Distributed Computing:** Many organizations are just realizing the cost advantages of establishing a network and desktop infrastructure. Unfortunately, what is often neglected are the components required to support the smooth operation of networked applications within the infrastructure. New applications are frequently introduced into a business environment without fully considering what their impact will be on the existing infrastructure. The result is less-than-optimal performance.
- **Problem Isolation:** When problems do occur in a distributed environment, it's difficult to isolate the problem to its source, hampering troubleshooting and fault resolution. The weakest link in the chain defines the entire transaction, and finding that link is crucial to characterizing application performance.
- **Fingerpointing:** For many applications, successful operation requires resource commitments from multiple groups. It is not unusual to have three or four major functional areas play a part in maintaining a networked application. The lack of a single, accountable owner makes it difficult to identify and fix an operational service level performance problem with these applications.
- **Lack of Service Level Targets:** Given the complexity of the networked application environment, many organizations have ill-defined targets. Such targets are difficult to establish, let alone enforce in a uniform manner. It is vitally important to achieve a consistent service level model across all applications so that a single investment in management policies and practices can be leveraged across the enterprise and applied universally to all applications.
- **Limited Visibility:** In many cases, there is a need for cooperative instrumentation from all vendors who are contributing to the overall networked application environment. Typically, application vendors will provide instrumentation and tools for their own products — often at the exclusion of other useful performance data. In other cases, application instrumentation is a low priority; to overcome this, traditional approaches will examine performance data from other locations. However, the recent widespread adoption of switching and remote access technologies has greatly reduced the visibility of these types of approaches.

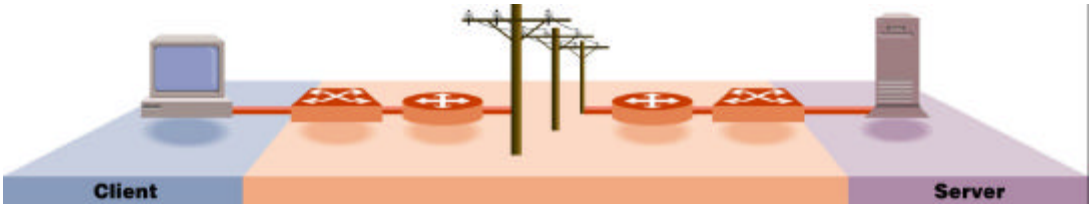

**Figure 2. Typical End-to-End Client/Server Patch,
in which the User Is Accessing a Central Business Application**

Defining Application Management

The field of application management is a complex and challenging one. In fact, the term "application management" itself is a bit misleading; the phrase is actually more generic than specific, since it encompasses so many areas and touches on so many different aspects of what an application does and how it does it.

This document focuses on one of the more compelling trends in the application management arena: application performance management. Distilled down to its most basic elements, "application performance management" can be defined as the pursuit of consistent, reliable *performance* and maximum *availability* — two issues that have the greatest impact on businesses that rely on applications to support their day-to-day operations.

Delivering consistent, reliable, and highly available application performance requires monitoring the active application — as well as its environment — within the context of business transactions in order to determine how it is behaving. If a problem is detected, application performance management also requires taking preventive or corrective action to ensure consistent delivery of services to users. Additionally, proactive planning comprehensive trend analysis and the establishment of performance baselines is required to proactively plan network growth and deployment.

This philosophy of managing application performance and availability is not new; it's the same concept employed by several "host-centric" or "device-centric" management solutions. Unfortunately, the vast majority of application performance problems — particularly in today's complex distributed environments — are not related to a single component or device. Instead, most application performance problems are the result of numerous complex interactions between multiple components. Even the subtlest event on a single device can have profound effects on application performance, especially when compounded by similar and equally subtle events on numerous other devices. As a result, each individual device or component can appear to be functioning normally according to the device-specific management solutions. But the net result for the application is less-than-satisfactory performance, which, for users who depend on that application, is the most important consideration of all.

One contemporary approach that addresses this issue is the Application Response Measurement (ARM) application programming interface (API) sponsored by Tivoli and Hewlett-Packard. The ARM API enables distributed applications to provide management platforms with critical business transaction information from a business operations perspective. While ARM represents an excellent method for monitoring distributed applications, historically applications must be modified in order to work with the standard. What is needed is a method for measuring business transactions and associating those measurements with the specific components involved in the delivery of those transactions — without requiring custom alterations.

The Nomenclature of "Application Performance Management"

What exactly is "application performance and availability"? What yardstick is applied to measure and quantify two such seemingly subjective terms? The following is a brief description of application performance management and application availability management, including the parameters used to define and measure each.

- **Application Performance** is usually quantified in terms of *Response Time* or *Transaction Time* and *Throughput*. The extent to which these two factors define an application's Quality of Service levels can vary depending on measurement parameters, but it is critical in applying relative importance using well-

established service level criteria. For instance, application response or transaction times that extend beyond a certain limit are typically nothing more than a minor annoyance to the user. In some cases, however, excessive response or transaction times can cause the user to abort the session, which may result in a significant loss of productivity. And in extreme cases, long response or transaction times and the subsequent aborted sessions can actually corrupt system resources, resulting in a very expensive recovery process. Therefore, two significant components of response or transaction times and throughput management are the accurate recording of performance and the ability to manage service-level thresholds on an application-by-application basis.

- **Application Availability** is another important aspect of application management. Sometimes, due to several factors, the advertised services of a networked application are simply unavailable to the end user. These conditions are often categorized as "hard failures." When such hard failures occur, the management solution must record the individual event with enough detail to facilitate problem isolation. Additionally, the solution must provide a meaningful way of aggregating individual failures into a larger picture distilling the measurements into concise reports of availability. This helps put the failures within the proper context, such as the percentage of successful transactions. It's also important to consider application hard failures within the context of organizational policies such as scheduled down time, LAN, WAN and dial-up situations, etc.

When a problem does occur that reduces application availability, it's important to resolve it as quickly as possible. Problem determination and resolution consists of a three-step process: first, the capability to detect it; second, the ability to locate the problem and isolate it to a specific location within the system; and third, the ability to determine how to correct it. Knowing which transactions are running at the moment the problem occurs, and understanding which components of the application or network infrastructure are involved in the transaction, takes much of the guesswork out of the situation. However, knowing this information requires visibility into the transaction — clear insight into what is happening between the user and the application server.

Determining the Best Vantage Point to Monitor Application Performance

As mentioned earlier, in order to effectively determine application performance and availability it is necessary to monitor the active transaction of an application, as well as the application's environment. The best place to perform this transaction monitoring — the vantage point that offers the most valid, most accurate view of application performance — is a subject of great debate among vendors competing to develop an effective application performance management solution.

In today's distributed client/server environments, there are actually several locations along the end-to-end network path between the user and the server from which to monitor and characterize application performance. Each has its advantages and disadvantages, and the accuracy of the measurements — as well as the resources and costs required — vary with each location.

The Server

One of the most obvious locations for monitoring application performance is at the server itself. Its central location makes it a very convenient solution for both deployment and administrative purposes. The server offers a strong application-specific perspective and detailed diagnostic information. In addition, if the networked application is multi-tiered, instrumentation at intermediate "server" locations provides visibility into these components. And monitoring at the server means relatively few components to instrument, which makes it logistically desirable.

However, the sheer volume of transactions makes instrumenting the server a computationally demanding task. Servers are mission-critical devices; adding the burden of processing and instrumenting literally thousands of additional transactions creates the risk of degrading performance and reliability, which runs counter to the objective of application performance management. In addition, the large number of hardware configurations and platforms, disparate operating systems, and the complexity of the server environment makes it difficult for application performance vendors to develop and provide a cost-effective solution. And the centralized nature of the server means it lacks visibility into network or user experiences and their correlation to business transactions — a serious shortcoming.

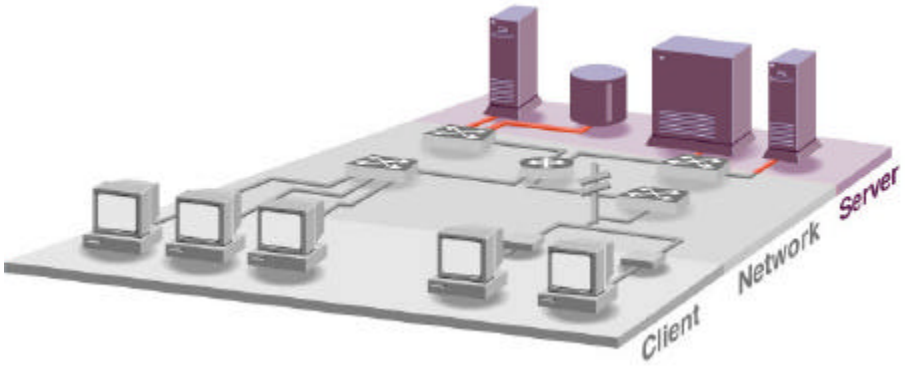

Figure 3. Monitoring Application Performance from the Server

The Network

Another potential location for monitoring application performance is from the network itself, either at a location near the server, a location near the client, or any arbitrary place in between the client and the server.

Promiscuous monitoring devices such as RMON probes and analyzers already support application traffic monitoring from the network. In fact, probes and analyzers offer strong Network- and Data Link-layer diagnostics, making them excellent tools for troubleshooting network and device-centric problems. And while some RMON2 probes provide additional visibility into Application Layer traffic statistics, such data is typically not presented in the context of business transactions.

The biggest factor working against promiscuous monitoring devices is the proliferation of technologies such as switching, remote access, and VPNs in today's distributed environment. Probes and analyzers rely on broadcast or shared media solutions such as Ethernet in order to work, and the point-to-point solutions employed in networks today render these tools effectively obsolete.

Probes and analyzers also share a drawback with servers: complexity. The sheer volume of transactions and conversations traversing a network path can easily overwhelm a standalone analyzer. Coupled with the fact that probes and analyzers — which are still fairly expensive — monitor basic low-level end-to-end

conversations and not business transactions, it becomes clear that monitoring application and transaction performance from the network, is not the answer.

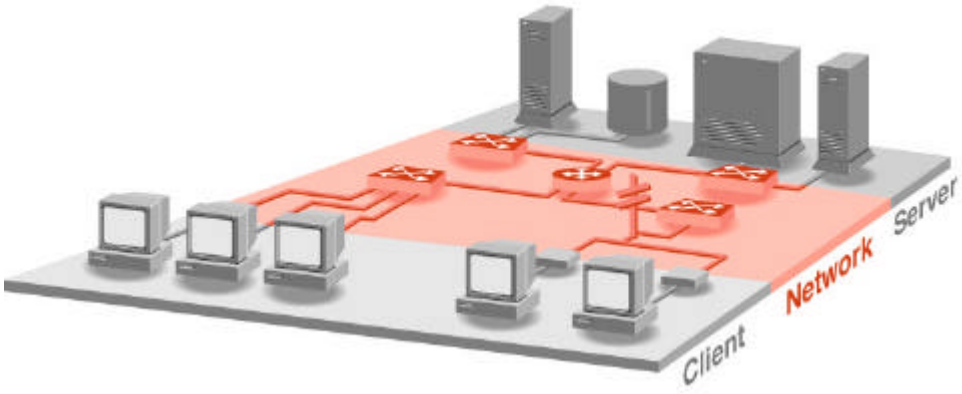

Figure 4. Monitoring Application Performance from the Network

The Client

The final location for monitoring application performance is the client's desktop. While there are challenges associated with the client (specifically deployment of instrumentation and the lack of visibility into intermediate levels of multi-tiered networked applications), the advantages of the desktop as a location for monitoring application performance far outweigh the drawbacks.

First, monitoring application performance from the client side introduces a "divide and conquer" mentality that eliminates the risk of overwhelming a single system device with too much monitoring data. Rather than expect a single server or probe to analyze thousands of transactions, the client-side approach utilizes hundreds or thousands of desktops to process a relatively low number of transactions each, distributing the burden throughout the enterprise. Because analysis of TCP and application flow is restricted to the single path that is being observed, it is possible to collect and process much more granular and powerful information regarding the entire end-to-end environment.

Second, because the desktop represents the place where the vast majority of application transactions begin and end, it's possible to perform a complete analysis of the end-to-end path, from the desktop, through the network, to the server and back again. As a result, it becomes possible to determine transaction and response times exactly as experienced by the end user — the ultimate service-level index.

Third, the desktop vantage point opens a number of possibilities for instrumenting client/server interactions and for combining data (transaction fingerprints and signatures) into a complete end-to-end performance and availability service level analysis.

The client location offers several other benefits as well: it represents the optimal vantage point for characterizing information flow from server to desktop. Data analysis is made available to the end user in a timely fashion. It allows correlation of client-side interaction by pairing network layer segments with the corresponding application-layer send and receive stream. And because it leverages the existing desktop, it's an economical approach that requires no additional hardware investments.

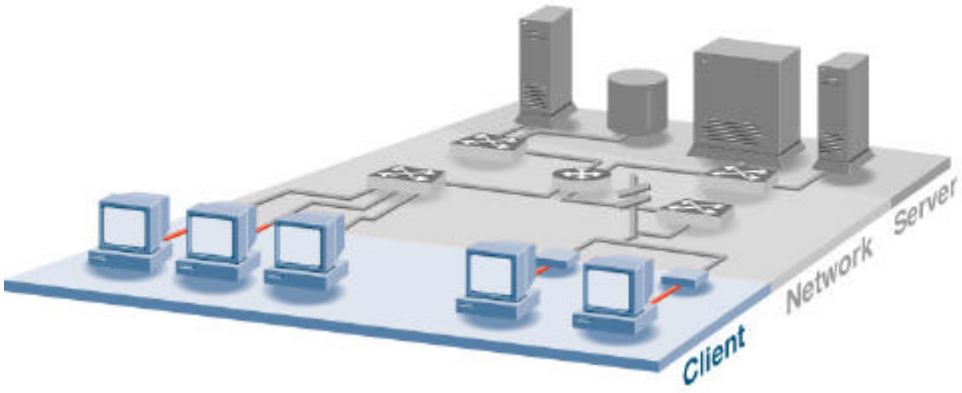

Figure 5. Monitoring Application Performance from the Client Desktop

The Most Valuable Vantage Point

Clearly, a case can be made for monitoring and measuring application performance at each location — the server, the network, and the client. And in most environments, there will typically be at least two, because system administrators and IT professionals have a need for the unique perspective offered by these different vantage points. But in terms of performance, scalability and economics, the client emerges as the most desirable location for characterizing true end-to-end application performance and availability in a client/server environment.

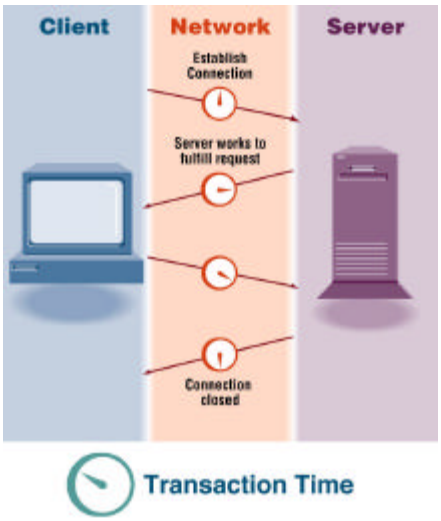

Figure 6. The Desktop Provides the Most Valid and Accurate View of Application Performance

Why the Desktop? Why Now?

Despite its numerous benefits, characterizing application performance and availability from the client desktop has not been possible until recently. The absence of a standard desktop environment, a shortage of processing power, and disparate network operating systems all worked against the client view. But the recent convergence of three simple yet significant events has turned the desktop into the most consistent and predictable location within the IT infrastructure for managing end-to-end application performance.

The first development that hastened this capability is the homogeneity of the desktop, thanks to the Microsoft Windows operating system. The popularity and proliferation of Windows machines has established a de facto standard for the desktop environment, which in turn has opened the door for a consistent method of monitoring activity from the client. A related, and no less significant, development is the growing robustness of the 32-bit Windows platform (both application isolation and base services such as multitasking) and its ability to support mission-critical business applications. Increasingly, as more demanding business processes are implemented, this robustness is almost completely taken for granted.

The second, and perhaps most influential, factor that enabled end-to-end application performance monitoring from the desktop is the widespread adoption of TCP/IP and routed backbones as the accepted standard for network operating protocols. In the past, the myriad protocols used in networks around the world — from AppleTalk to NetWare — created a very complex environment for monitoring and analyzing network performance at the desktop. Today, TCP/IP has created a consistent network environment for capturing the vast majority of network communications, which in turn enables end-to-end performance analysis throughout the entire enterprise.

The third development that made client-side monitoring possible is the impressive processing power now available on virtually every desktop, compliments of Intel. Moore's Law states that the capacity of the computer chip will double every two years, and recent advances have borne this out, putting more processing power on the desktop than ever thought possible. Monitoring, measuring and analyzing individual network transactions takes some processing power, even when restricted to transactions related to an individual client. Thanks to the ubiquity and power of today's chips, even the most basic laptop computer features enough CPU cycles to passively monitor application performance without impacting other routine tasks.

Today's IT infrastructure includes a number of critical components: networks, servers, applications, and desktops. But these three factors have driven a convergence towards a "universal desktop" standard that has made it possible to monitor and characterize application performance and availability from the most valuable and valid perspective there is: the end user's. Thanks to the new network environment, it is possible to leverage existing resources and computing investments to observe how applications are performing for those information workers who spend their days online to keep business running. By measuring application performance at the desktop, where transactions begin and end, network administrators and IT professionals gain complete visibility into the unique end-to-end paths that define network and application performance.

The VIA Architecture: Visibility Into Applications

Now that it is possible to characterize application performance and availability from the desktop, it makes sense to develop a method that takes advantage of the data available through this valuable resource.

Developing such a method requires an architecture that defines a specific method for monitoring application performance and availability from the desktop. There are a number of sources of performance information at the client desktop, and a number of different techniques for monitoring, measuring and instrumenting them. The architecture should address the challenge of capturing, aggregating and presenting vast amounts of information in a meaningful and useful way while addressing the issues surrounding deployment and configuration of agents on individual desktops.

VitalSoft, the software division of Lucent NetworkCare, was the first to develop such an architecture — a comprehensive blueprint that defines a method for managing network and application performance and availability by instrumenting the client's desktop. This white paper discusses what makes the VIA

architecture unique and how these techniques deliver unparalleled, end-to-end visibility into networked applications.

The VIA (Visibility Into Applications) architecture defines an open and extensible framework for monitoring and analyzing end-to-end application performance from the desktop, through every device on the network path, to the server and back again. By taking advantage of the spare resources available on user desktops, as well as exploiting the new network environment to provide a vehicle for collecting the data, VIA provides unprecedented visibility into application performance and availability — everything from e-mail to database and business applications.

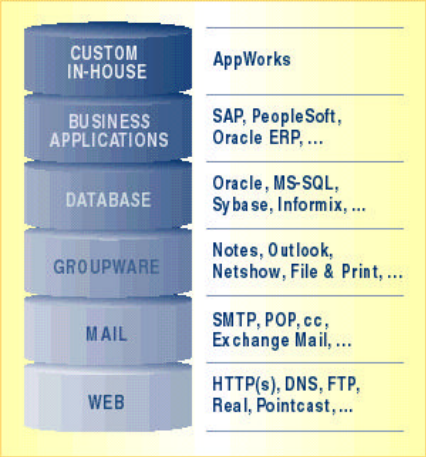

Figure 7. The VIA Architecture Supports Business-Critical Applications

Desktop Agent

The VIA architecture is based on three fundamental technologies — passive flow analysis, transaction analysis, and AppWorks™ — that deliver the ability to not only monitor each individual application transaction, but also to dissect them in granular detail. These technologies, unique to VitalSoft, are incorporated into VitalAgent™ — a common, standard desktop agent that performs the data collection and analysis. From its location on the desktop, where it performs root cause analysis, data aggregation, intelligent filtering and threshold management, VitalAgent represents a scalable solution that easily adapts to the largest enterprise networks.

Centralized Correlation and Aggregation

The data collection, analysis and filtering performed by the desktop VitalAgent is only half of the VIA architecture approach. That data is then forwarded to a central collection point, where it is correlated and aggregated to provide the IT professional with a comprehensive overview of application performance and availability. That centralized data collection, aggregation and correlation is performed at the VitalAnalysis™ and VitalHelp™ server components — two products that, along with VitalAgent, are part of the VitalSuite™ product family. The following diagram illustrates how the VitalAnalysis and VitalHelp server components interact with the VitalAgent component.

In addition to aggregating and correlating the data forwarded by VitalAgent, the VitalHelp and VitalAnalysis server components also provide mechanisms for centralized configuration, administration and deployment of VitalAgents throughout the enterprise. This role becomes critical as the scale of VitalAgent deployment increases.

The following sections describe, in greater detail, the architectural elements of VitalAgent that make client-side instrumentation a viable option for today's application-dependent environments.

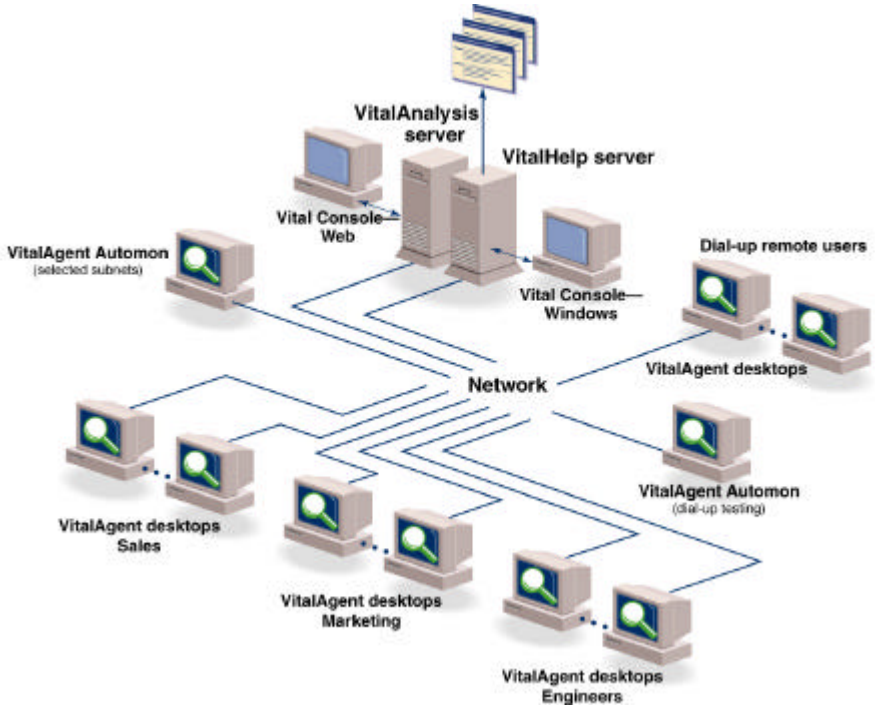

Figure 8. VitalAgent and VitalHelp/VitalAnalysis Deployment

Passive Flow Analysis

VitalAgent employs a number of instrumentation techniques — including TAPI, Windows messages, and other advanced technologies — to passively gather performance information from the client desktop. One of these techniques, passive flow analysis, deserves further discussion, since it is the critical component in providing visibility into the performance of the complete end-to-end path — from the client, through the network, to the server, up to the Application Layer and back again.

The design, implementation and deployment of TCP/IP-based networks have reached a very mature stage. Most TCP implementations behave in a generally predictable way during connection establishment, data transfer, and connection tear-down [Stev94a, Pax97a]. In general, TCP implementations push the network to utilize the available network bandwidth capacity in an efficient manner. It reacts to signals such as receiver buffer exhaustion, network congestion and current network round-trip-time. Good implementations use the available network capacity and deliver data to receivers in a timely way while keeping network overhead to a minimum.

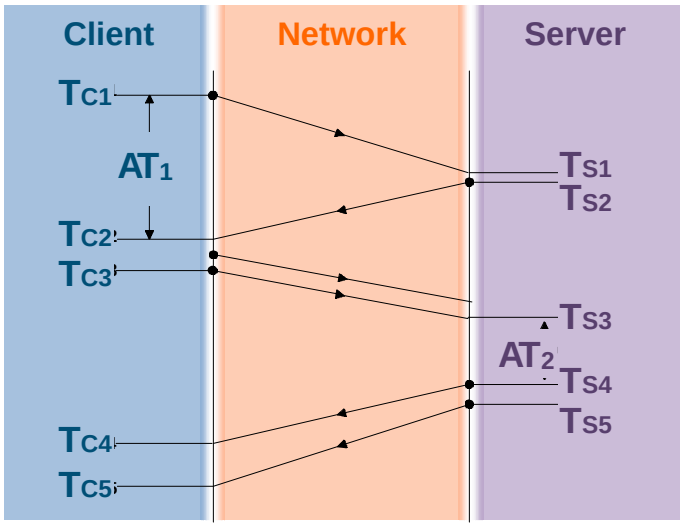

Figure 9. Passive Flow Analysis Illustrated

Passive flow analysis calculates and presents performance estimates based on monitoring the flow of data between the client and the destination server. For the purpose of discussing these measurements, we define a *session* as a series of TCP connections that the client establishes in order to complete a task, such as retrieving a document. Passive flow analysis monitors the entire session to provide a complete and accurate picture of application performance without introducing unnecessary traffic onto the network.

By observing TCP inter-packet dynamics such as packet inter-arrival times, slow start dynamics, packet bunch mode and other notable events, it is possible to gather information from the entire end-to-end path and gain visibility into network and server performance as well. This sophisticated analysis technique that harvests information inherent within the design of TCP/IP not only provides visibility into the application's performance, it also provides visibility into the network and server performance as well. It is this powerful technique that gives IT professionals true end-to-end visibility into all the components that make up a transaction.

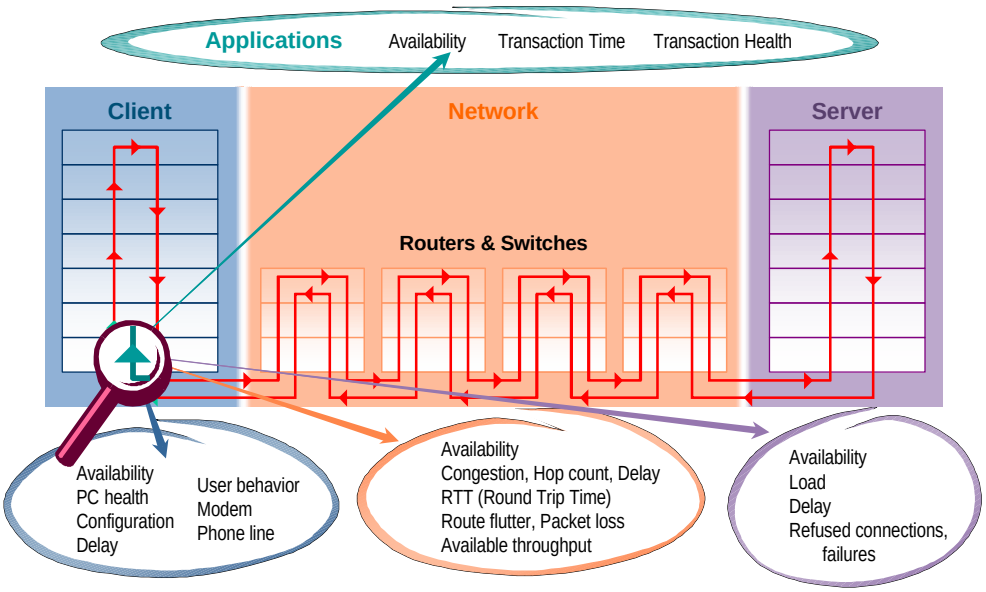

Figure 10. End-to-End Path Visibility with Passive Flow Analysis

Transaction Analysis

The second fundamental technology of the VIA architecture is transaction analysis — the act of analyzing each and every application transaction and breaking it down to isolate and identify the source of performance problems. The VIA architecture's transaction analysis technology provides a framework for monitoring the performance of individual transactions within an application. For example, when you use email, you "log on," "retrieve" and "send" messages. When considering the end-user perspective of these transactions, it is critical to identify the business transaction and apply the service level requirements associated with each. The VIA architecture recognizes the importance of characterizing not only the application but those transactions that are critical to the business.

Learning Engine

By tracking critical application performance indicators at the client location, VitalAgent learns the "signature" of each application or each transaction's progression for the unique client/server environment. This intimate history of inter-packet dynamics and past performance, combined with the stochastic models for application performance, allows VitalAgent to "learn" the signs indicating degrading health. This information also provides a basis for comparison to recent application history — for example, dramatic degradations beyond what has been seen previously — to create an immediate event. This addresses the problem of trying to set a global threshold for a critical application parameter that varies dramatically across different environments and creates a more intelligent framework for problem demarcation and analysis.

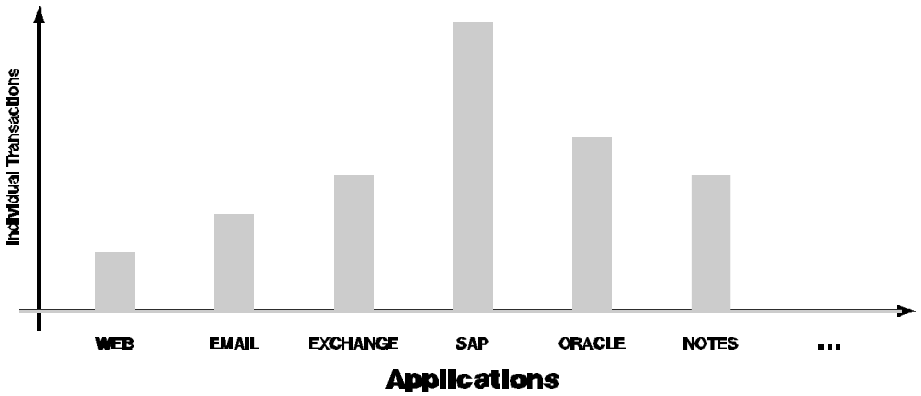

Figure 11. The VIA Architecture Supports Applications as well as Unique Transactions

Stochastic Techniques

Most transaction measurements do not follow conventional statistical distribution (such as Poisson or Gaussian) [Pax97]. As a result, mean latency is rarely an accurate indicator of a component's normal or "typical" behavior. A more appropriate measure of normal behavior is the median. If the complete distribution of a transaction is available, the percentile of a sample can be used to determine the probability that an observation of this size or less would be seen in practice.

VitalAgent tracks all critical parameters locally and maintains distributions of each parameter. Additionally, the process of aggregating these parameters maintains the complete distribution. This capability allows determination of various measures of dispersion such as median, mean, inter-quartile range and standard deviation. The result: a far more detailed and accurate view of what's really happening with application performance.

Demarcation of Performance and Availability

One of the key benefits of the VIA architecture's transaction analysis is its ability to pinpoint or demarcate application performance and availability problems. This capability helps to end the "fingerpointing" that occurs in many organizations when trying to determine the source of application problems. By applying root cause analysis to raw measurements, VitalAgent determines the component that is the point of failure. In addition, the correlation and aggregation components on the VitalHelp and VitalAnalysis servers further isolate the problem by attributing it to a specific source.

One example of this capability is reflected in the way VitalAgent characterizes transaction and response times into their respective components: client, network, and server. This allows IT organizations to quickly determine where to focus their resources or make additional investments to improve application performance. Is the problem with the server, or is it isolated to a specific application issue? The answer will determine whether a planned network upgrade is really necessary. By isolating the source of application performance problems, the VIA architecture and VitalSuite products help IT professionals make intelligent, informed decisions that can have a direct impact on the bottom line.

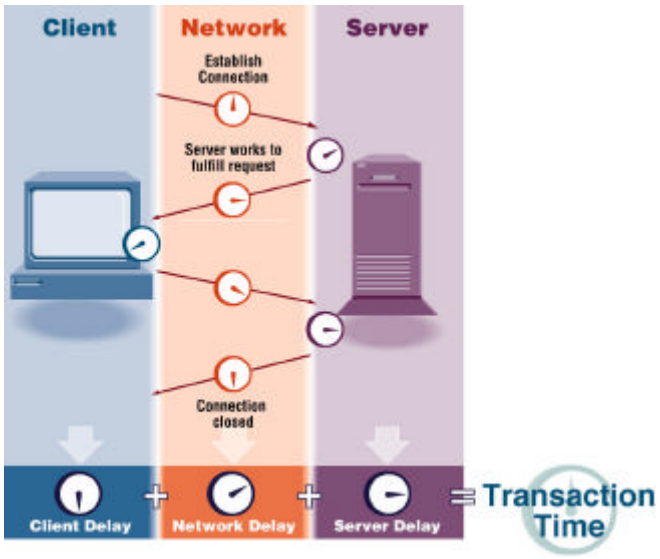

Figure 12. Transaction Analysis — Client Delay + Network Delay + Server Delay = Transaction Time

Or consider the case of a networked application that relies on DNS name resolution. In most situations, the system's TCP/IP configuration allows the specification of primary and secondary DNS servers. In cases where the primary server is inaccessible or fails to resolve a name, the secondary server is contacted. Let's assume for a moment that the secondary server is accessible and is able to resolve the name. While the transaction completes successfully, the user would perceive a large delay. By correctly associating the underlying failure with the primary DNS, the VitalSuite products provide reveal this performance degradation to provide insight into network operations.

Passive Flow Monitoring vs. Synthetic Transactions

One technique that IT organizations frequently employ for performance characterization is to deploy software agents that generate a sequence of transactions designed to emulate a user's behavior. This "synthetic transaction" approach offers a valuable benefit: the ability to create a repeatable and controlled environment for both baselining and trending purposes.

VitalAgent Automon™ — an advanced version of the VitalAgent software — provides a mechanism for automatically generating application-specific synthetic transactions to perform baseline and trending analysis of corporate resources. Working in concert with VitalAgent, VitalAgent Automon makes it possible to create an "artificial" environment that tests and validates the results that VitalAgent is reporting. Other third-party products such as SQA Robot and LoadTest (from Rational Systems) and WinRunner (from Mercury Interactive) can also be deployed to complement VitalAgent's session and transaction flow analysis.

While synthetic transaction generation is attractive, it does have a few pitfalls. For instance, in many situations application failures are the result of incorrect or poor configuration of server components. Such configuration problems are unique to each desktop, and it is vitally important to get visibility into such issues — something that these artificial workload generators don't provide. Another problem is the significant amount of network traffic and/or load that is placed on the application and database servers — unnecessary traffic that could threaten overall performance.

ARM: The Industry Standard for Performance Management

The Application Response Measurement (ARM) API, mentioned earlier in this document, provides an open framework for application vendors to export performance data of business transactions to other management platforms. ARM has evolved into two major versions, the first focusing on baselining instrumentation for transaction demarcation and the second building on top of the first to introduce user-defined metrics and main and sub-transaction correlations.

The VitalSuite products support the ARM standard by providing an *ARM proxy* that allows integration with leading management platforms such as Tivoli's TME and HP's OpenView. Furthermore, VitalAgent takes full advantage of ARM 2.0 capabilities to map and export several key transaction-related performance measurements using the user-defined metrics capability.

With the ARM proxy, VitalAgent extends the ARM standard to many business applications that are not yet ARM-compliant. That means organizations using those applications get all the benefits of ARM without having to instrument, modify or otherwise alter their applications to work with the standard. This capability extends the VIA approach to characterizing performance for packaged as well as custom in-house applications, creating a standard approach that protects existing investments.

Additionally, with systems management products from vendors supporting ARM (including Tivoli TME10 and HP's MeasureWare/PerfView), the performance data from VitalAgent becomes available immediately.

AppWorks: Extensibility into New and Custom Applications

Increasingly, organizations are developing in-house applications based on a distributed computing architecture to meet their specific businessneeds. While often more effective than off-the-shelf solutions, these in-house applications are typically high maintenance solutions that require a great deal of hands-on management to perform properly. And without the support of third-party management tools, organizations find it difficult, if not impossible, to cost-effectively provide the administrative support these in-house applications demand.

VitalSoft addresses this shortcoming with the AppWorks capability, a fully open approach that allows organizations to customize the VitalSoft solutions to meet the unique performance management needs of their custom in-house applications. Featuring the ability to extend management support to new, non-standard and other unique applications, AppWorks offers the flexibility to change application performance management capabilities as the organization evolves. Furthermore, organizations benefit by a uniform application of service levels across both the in-house and off-the-shelf applications.

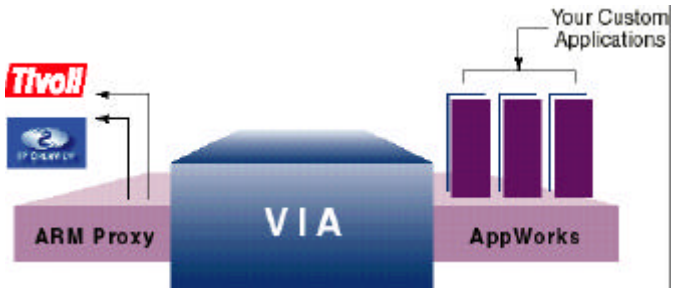

Figure 13. The AppWorks Capability Allows VitalSuite to Support Custom In-House Applications

AppWorks Template

This custom support is accomplished in one of two ways. The first method for supporting unique in-house applications is through the AppWorks Template, a capability that enables VitalSuite to monitor custom in-house applications that run over TCP/IP. This easy-to-use template provides rapid transaction-level support for new applications. A centralized web-based form allows quick and easy definition and deployment of these custom in-house applications.

The AppWorks template extends basic VitalSuite capabilities to any TCP-based application. By applying the AppWorks Template, IT professionals can obtain transaction summary performance and event information for their custom applications, all from the end-user perspective.

Defining custom applications for the AppWorks Template is a streamlined process. The definition is captured in a concise form, which is then distributed to VitalAgents on user desktops. As soon as the definitions reach VitalAgent, the software immediately begins tracking the newly-defined applications, reporting performance data back to server components. By defining applications using the AppWorks Template, users can view reports through VitalAnalysis and track events through VitalHelp, where VitalSoft's built-in root cause analysis, problem isolation and automatic troubleshooting capabilities will further diagnose the problem and record contextual information for use by an IT professional.

Scalability: Distributing Intelligence to VitalAgent

In order to scale the VIA architecture and the VitalSuite product family to support the needs of large enterprise environments, VitalSoft endeavored from the earliest stages to distribute intelligence to the desktop VitalAgent.

VitalAgent is able to run autonomously (i.e, with no console connection) because it is essentially self-contained. All of the application discovery, software and configuration updates, as well as all the centralized policy and management rules the agent needs, are self-managed by the agent itself. Much of this intelligence is contained in the "plug-in" Knowledge Module libraries that VitalAgent autonomously updates. From policy and configuration to local administration and data repositories, VitalAgent tackles a number of tasks that have historically been handled at a centralized server.

The multi-threaded agent often monitors hundreds of parameters on each application, interprets the data, stores the data locally in history files, and implements predetermined corrective actions. Because the agent is autonomous, it can do all of this without any communication with or connection to a centralized management console. This degree of autonomy is critical for creating a scalable and robust management environment and produces significant gains in resource efficiency by reducing the amount of information that must pass between the agent and the manager.

This distributed philosophy enables the VitalSuite family to scale to the needs of large enterprise environments. After all, the server-based VitalSuite products do not have to maintain the information on the managed objects, making it a very resource-efficient solution. Using "exception-based" monitoring, only those events that fall outside of defined norms and require administrative attention are forwarded by the agent to the console, dramatically reducing related management traffic and central data repository requirements.

VitalSoft's separation of the agent and the server-based manager applications contributes to its impressive scalability. Other management products and frameworks that rely on the console for threshold determinations, recovery actions and archiving are quickly overwhelmed in larger environments, restricting their performance and reducing their reliability

Openness

The VIA architecture was designed as a flexible and open architecture — an extensible approach that ensures longevity and allows users to leverage their existing management investments, not replace them.

The VIA approach integrates with products from leading systems and application management vendors. As described earlier, by using the ARM API, the VitalAgent exports transaction performance data to these solutions. For added convenience and flexibility, all VitalSuite server products are HTTP-enabled and can be accessed via a Web browser from any desktop. Communication between VitalAgent and the server-based manager is HTTP and HTTPS, supporting popular e-commerce, VPN, and collaborative management requirements.

Another example of the VIA architecture's open approach is the way in which the VitalHelp component of the VitalSuite product family integrates with leading help desk solutions such as Remedy ARS. VitalHelp is fully integrated with the Remedy ARS trouble ticketing system, allowing events discovered by VitalHelp to be automatically integrated into the ARS system for assignment and tracking. When a trouble ticket is closed in Remedy, it is also closed in VitalHelp, making it easy to identify, locate, track and resolve application performance problems — without anything slipping through the cracks. Additional third-party platforms and applications can also be supported through an External Event Gateway, providing more evidence of the VIA architecture's open design.

VIA is open from a reporting and data extraction perspective as well. By implementing a standard relational database in the VitalSuite product family, VitalSoft has made it possible to customize reports and consolidate data with minimal programming effort. VIA also uses standard ODBC-based database query access and provides the database schema for developers, so this data can be incorporated into business decision support systems such as an OLAP cube product.

Summary

As networked business applications become more critical to daily business operations, IT professionals demand a method for monitoring and managing those applications to ensure maximum performance and availability. The VIA architecture, embodied in the VitalSuite product family from VitalSoft, delivers such a solution. By analyzing network activity from the desktop, where every transaction begins and ends, the VIA architecture provides the ultimate end-to-end service level index for application performance: the end user's experience.

The unique approach defined by the VIA architecture leverages the existing desktop investment to deliver an aggregate view of each end-user's total application experience. VitalAgent software on individual desktops, utilizing patented transaction, passive flow and root cause analysis technologies, collects the data and reports it back to centralized VitalHelp and VitalAnalysis applications, where it is assembled into a comprehensive overview of network and application performance. The result is a complete and accurate understanding of how mission-critical business applications are behaving for those who rely on those applications to do their jobs — without requiring costly application changes or expensive, dedicated probes.

The VIA architecture and VitalAgent also deliver a self-help capability to the end users themselves, lowering total cost operations while improving returns on investments for the management system. By providing users with a high-level analysis of encountered problems, as well as a recommended course of action for solving the problems, VitalAgent helps keep support phones on the hook, enabling IT professionals to concentrate on more widespread problems.

By virtue of the current desktop environment, the VIA architecture provides something no other vendor has been able to offer in the past: a common and consistent framework for managing application performance and availability. With its broad support for existing TCP/IP-based applications, VIA delivers a solution that is equally suited for off-the-shelf business applications as well as custom-developed in-house applications. Designed from the outset as an open, standards-based approach, the VIA architecture and the VitalSuite product family also integrates with and complements existing systems management solutions, while support for the ARM standard provides access to other new and custom applications that support the standard.

The VIA architecture represents a breakthrough in application performance management: a new approach that solves today's management challenges quickly, easily and economically.

For More Information

For more information about the VIA architecture or the VitalSuite product family, please visit our web site at http://www.vitalsuite.com.

For further information, see the NetworkCare web site at http://www.lucent-networkcare.com or call 1-888-767-2988 in the U.S. or 1-727-217-2303 outside the U.S.

Copyright © 2000, Lucent Technologies NetworkCare

This is an unpublished work protected under the copyright laws. All rights reserved.
WP.VS.VIA.0100