Maximizing SQL Server Application Performance and VERITAS i³

USING VERITAS i³™ FOR WEB APPLICATIONS RUNNING MICROSOFT SQL SERVER
TABLE OF CONTENTS

Application Performance Management ................................................................. 3
Executive Summary ............................................................................................... 3
Introduction ........................................................................................................... 4
The Typical Environment .................................................................................... 4
Challenges Managing the Environment .............................................................. 5
The VERITAS i3 Methodology ............................................................................ 8
VERITAS i3 for Web Applications ...................................................................... 10
  VERITAS i3 for Web Features ......................................................................... 12
  Proactive Management through Exceptions .................................................. 12
  End-user experience and SLAs ......................................................................... 13
  Correlated end-to-end view ............................................................................. 14
  In-Context drill down ....................................................................................... 15
  Expert In-depth Analysis ................................................................................ 16
  Expert Recommendations .............................................................................. 19
  COM+ Correlation .......................................................................................... 19
  Long-term Analysis ......................................................................................... 20
Methodology ....................................................................................................... 21
Benefits of VERITAS i3 for Web Applications .................................................. 22
Summary ............................................................................................................. 23
EXECUTIVE SUMMARY

Web-based applications have become an essential component of every business in order to compete and realize the necessary savings and productivity gains demanded to achieve their bottom line objectives. Their deployment has dramatically changed the fundamental economics of service delivery, competitiveness and customer or employee interaction. Increasingly web-based applications are replacing the more expensive and less responsive traditional “people assisted” services.

These new web applications exploit existing investments and leverage new technologies such as new application servers (e.g., Microsoft .NET and COM+ application servers, BEA WebLogic Application Server, IBM WebSphere Application Server) to meet time to market requirements. However, the combination of multiple tiers, distributed processing, new technologies and the reliance on a public network have significantly stressed the existing portfolio of performance management tools. The more tiers the more tears, Web-based application infrastructure is complex - paradoxically, these web applications actually demand more responsiveness and around-the-clock high performance than their less complex internal predecessor applications in order to attract and retain the client base. As such, many implementers are experiencing a very high rejection rate, even from internal users, after a bad experience. The rejection manifests itself in terms of lost customers and unrealized savings for the application as existing customers revert to the more traditional and more expensive service interfaces.

Effective web applications implementers recognize the need to include application performance best practices throughout the entire application life cycle from development to deployment and operation. A critical aspect of web application best practices is recognition that many things contribute to performance overhead. One of the biggest challenges is measuring the actual service being delivered to the end user (referred to as the user experience) and then being able to determine the end-to-end service level contributors. These measurements are needed in order to make the appropriate tuning, design or architectural changes.

Understanding how the actual service being delivered impacts the business is a key factor. What is the business impact of an application slowdown? Does it affect all organization users or just specific users at specific locations? Does it happen at critical business peak hours or off hours? What is the business impact of not meeting organizational SLAs? How could we leverage the existing application infrastructure to better service the business?

The biggest single challenge facing IT today, when it comes to web applications, is finding a repeatable and understandable way to measure the service level of each application and compare it to the business objectives. VERITAS i^3 for Web Applications is the industry’s first software solution that provides a way to quickly, efficiently and unobtrusively capture the hard to get URL-to-SQL application metrics. It presents these important metrics in a manner that enables crisp communication, rapid proactive or reactive detection, correction and verification throughout the application life cycle. This paper describes this unique approach.
INTRODUCTION
The web has enabled many companies to realize significant saving and improved responsiveness by off-loading traditional in-house service interfaces to their customers or prospects while expanding the company’s reach and business hours availability. As such electronic service, electronic delivery and self-service front-end applications leveraging the web and web evolved technologies have changed the shape of delivery systems, application architectures and even the systems they run on. The resulting web-based multi-tier, multi-server distributed processing systems running “web applications” have become the cornerstone or bane of many big and small companies alike.

User service level expectations have also evolved over this period as well; they now expect to be able to access your site at any time from any location around the world. They expect to access your applications in uncontrolled patterns and patterns which leverage various levels of caching in the desktop and the network. If your clients or prospects see that it takes too long for the blue bar on the bottom of their screen creep from left to right they simply leave and frequently won’t return. If the service level doesn’t meet the client’s expectation, existing clients often revert back to more traditional and expensive interfaces to deal with your company or start looking at alternative vendors or suppliers or services. Prospects simply move on.

The vexing part to companies providing services over the web is that they often can’t even see what the user is experiencing, can’t identify where the slowdowns are and many times can’t even detect when they have slowdowns and failures before the phone starts ringing. Equally challenging, once a problem has been detected, numerous organizations are called into action in order to isolate the friction point or points and then determine which group has the technical expertise to circumvent or resolve the problem.

Suffice it to say that as a result of these complexities most IT organizations and web application owners are on the lookout for performance management solutions that addresses their production monitoring needs and that can be readily used in the staging, quality assurance, performance/stress test and development environment. These organizations recognize the need to eliminate as many performance problems as possible before they go into production and then they need solutions that are useful in proactively and reactively recognizing: the symptom, determining the user impact, identifying the probable cause(s) and assisting in the identification of those friction points that have the highest pay-back.

THE TYPICAL ENVIRONMENT
A typical and “simple” web application environment might look like this:
The management of this “simple” application, spread across this multiple tiers, demands a comprehensive yet straightforward way to measure the user’s experience. Starting from the desktop, it must proactively assure that satisfactory service levels are maintained as expected. It must enable the proactive detection of service level deviations or reactively detect, alert, and document any performance issues.

For web applications the primary considerations are the service level provided to the user and the optimum configuration(s) to provide the required service level with “just-in-time” capacity as needed.

It is important to recognize that this new generation of web-based applications introduces several new performance considerations over traditional in-house applications using client server or mainframe implementations:

- The user not the application controls the flow
- The actual response times varies as usage patterns differ and the connectivity varies
- The multi-tier architecture of Web servers, application servers and database servers (e.g. IIS Web server, COM+ server and SQL Server) makes it hard to pinpoint the source of performance problems
- As more applications running on different server machines access the SQL Server database, database administrators would like to gain visibility to which server machines, COM+ applications and components access the database.
- Without correlating the COM+/ .NET application server access to the SQL Server database we don’t gain visibility to what happens between the two. COM+ components accessing the database are like “black boxes”. In order to solve slowdown problems and to understand the application impact we would like to identify which COM+ applications, components and methods access the database inefficiently, what is the impact of a long running SQL statement on the COM+ application and more.
- The multi-tier architecture may potentially add overhead to the response time
- Inter-dependence between tiers can result in problem cascading
- COM+/ .NET application environment inherently uses more “moving parts”
- COM+/ .NET applications are designed to run in parallel and in multi-processing configurations, designed to serve more users and transactions at various load levels
- Third party software packages and their interfaces increase the complexity of the overall environment

The new generation of Web applications adds new functionality as well as new tiers and components, which creates new kinds of problems and challenges to manage.

Key performance management requirements for this environment include the ability to promptly identify performance problems and put them in the application context, see the effect or impact on end-users using the application, isolate problem symptom and find its root cause and proactively analyze trends and predict future problems.

CHALLENGES MANAGING THE ENVIRONMENT
The management of Web applications, spread across multiple tiers, Web servers, application servers, middleware servers and database servers, makes it hard to pinpoint the source of a performance problem. It raises a few challenges, listed below.

- Ability to define, measure and commit to organizational business objectives and SLAs
- Understanding the business impact on which users and applications are impacted by a performance slowdown
- Ability to define normal and abnormal application and usage behavior through baselines
- Comprehensive yet straightforward way to measure the user’s experience
• Identify problem patterns and characteristics before the user is affected
• Proactively, through alerts, assure that satisfactory service levels are maintained as expected
• Proactively, through alerts, manage the system when deviation from expected normal behavior, or expected SLAs, happens
• Ability to track the performance of local and remote geographical locations

• 7x24, real-time collection and correlation of performance metrics across the application infrastructure, enabling problem isolation and root cause identification.

• Correlation of application metrics like user activated URLs, invoked COM+ objects, called database SQL statements across the Web application infrastructure. This provides the ability to isolate a problematic component, network, Web server, COM+/ .NET object or the SQL Server database.

• COM+ components accessing the SQL Server database are like “black boxes”. We would like to gain visibility to COM+ application and components accessing the database. The challenge is to correlate SQL statements and their origin COM+ methods that initiated these SQL statements. The motivation is to understand the application impact of problematic SQL statements and to identify which COM+ components access the database inefficiently and what is the impact of a long running SQL statement on the COM+ application.

• Ability to indepth analyze programs, users, SQL statements and objects at the SQL Server tier, find the root cause of the problem and suggest remedies.

• Ability to associate a logical SQL Server I/O problem with a physical disk access to identify bottlenecks

• Capture enough data for enabling the problem resolution. Enough data means a wealth coverage of data entities and for each entity relevant data metrics like service-times, number of invocations, throughput, wait time, etc.

• Ability to maintain historical data for trend, exception and capacity planning analysis for predicting future performance.

• Low overhead, continuous monitoring of application components, in a production environment

• Coping with unpredictable load periods and handling massive transaction/data volumes

• Aid application managers and administrators in analyzing performance issues and suggest remedies

These challenges are neither covered by system management frameworks, nor by synthetic robots testing the system from multiple locations, nor by stovepipe snapshots from various servers along the application path.

IT organizations with these kinds of configurations typically have systems management frameworks in place to monitor the performance and availability of the infrastructure and provide operations with a systems management interface. These frameworks use agents deployed in the network and servers to gather systems usage data about the various monitored components. This systems level data is monitored to raise alarms if the component’s utilization or average response time exceeds established thresholds or become unavailable. These frameworks are extremely well suited for and effective at detecting system level and network issues such as servers and applications with high utilization, high packet rates, and numerous retries. Their primary focus is monitoring the system and the system elements’ health.

System management frameworks are focusing on system parameters, availability and utilization. Their primary focus is monitoring the system and the system elements’ health. Therefore, frameworks extremely well suited for
and effective at detecting system level and network issues such as servers and applications with high utilization, high packet rates, and numerous retries.

Unfortunately system and element detail is not effective in providing a view of what the user sees at the desktop. Management frameworks are not focusing on end-user experience, activity and usage. The data can’t be used to effectively determine the problematic application end-to-end experience and friction points beyond generic issues such as the system or an element is over utilized or unavailable. Moreover, because of the multi-tiered architecture of these new web applications and new application technologies such as .NET and J2EE, the system level agents do not provide the detailed information needed to isolate the friction point for a specific application or user. Therefore if all system parameters look OK how can we tell for example why location X or certain users suffer from poor response-time, or why certain URLs are slow, or which COM+ objects suffer from long SQL Server access time?

Robots provide external samples of synthetic access scripts. It neither reflects the real load on the application and its distribution, nor the real transaction that are executed. Real application traffic could vary dramatically from a simulated one. For example, you have distributed robots at three locations, but today most of the traffic is coming form a fourth location, which you are not aware about. Another example, users suffer from bad response times for certain transactions while a robot monitors other transactions. Moreover, synthetic activity is activated every X minutes, which causes a gap in continuous 7x24 data collection of performance metrics. What happens between one synthetic sample and the next one?

The stovepipes approach, which takes snapshots form various servers like Web, COM+/.NET or J2EE components, SQL Server do not provide an answer to problem either. Snapshots provide very coarse information, generic statistics taken from each tier separately, no correlation across tiers. For example, one can look at Web URLs, COM+ objects or SQL Server metrics and still doesn’t understand which component causes the bottleneck for problematic transactions. Snapshots would tell you that the average URLs time or SQL response times are OK, but they cannot count specific invocation time for each URL or SQL statement or point out which SQL statement was invoked by each COM+ object, therefore cannot help much in isolating the problem.

Even the combination of management frameworks, synthetic robot and stovepipes is not good enough. Typical approaches to isolate end-to-end problems today use stovepipe detail snapshots of the performance of the various servers along a path outlined by the application specialist and data from geographically deployed robots. The robots initiate synthetic transactions to approximate what users might experience if they were to access the service using a certain network connectivity (e.g., 100Mbit LAN) from that location. While the system level information and representative user information helps with monitoring baselines or detecting deviations they do not provide sufficient detail or coverage to identify specific users or groups of users. The system level information does not account for the user’s experience at the application, transaction or page level. Without the critical correlated data the only option is to use highly skilled diagnosticians and a lot of time.

With the best of diagnosticians, without the critical information the problem determination to resolution process is based on educated guesses and the elimination of where the problem “isn’t” (e.g. a server has dropped from the configuration). But even the identification that a specific component in the application server has failed in a clustered J2EE or .NET environment, it may take an extended time to isolate because of the granularity and amount of activity.

Metrics for end-to-end application analysis must be captured with agents having application knowledge and visibility and those that reflect the users experience. Real-time performance metrics should be collected 7x24 and correlated across the Web application infrastructure path, enabling problem isolation and root cause identification.

Data collected from different tiers and application components must be correlated. For example, correlation of invoked URLs, J2EE or COM+ components and invoked SQL statement. Without correlated data it may take an
extended time to isolate the problem and find its root cause, because of the granularity and amount of activity and parallelism. For example, we have detected that all users executing a specific URL suffer from long response times. Now, where is the problem? Is it the network, Web server, application server, other middleware components or the database? Without correlating granular metrics across tiers we would not be able to solve the problem.

Collection of performance metrics 7x24 and correlating them is mandatory, but not enough for having a useful performance management solution, answering the challenges mentioned earlier. Historical information of collected data must be kept for long-term analysis, exceptions, baselines, trends and capacity planning. Based on historical data we can calculate the typical application behavior, for example there are peaks each Monday morning and at month end. Knowing the typical behavior we can generate exceptions for abnormal behavior or set the right SLA thresholds. Another example, based on historical data we can calculate the database table growth and allocate enough space in advance or create appropriate indexes to reduce the fetch times.

In short, typical infrastructure frameworks provide critical system and network management bottoms-up detail but do not look at end-user and transaction activity, robots provide external samples of synthetic access scripts, stovepipes look at each tier separately providing generic statistics for that tier, these approaches lack the power to find, isolate and focus on the root cause of the problem as well as the real user response time experience. 

VERITAS i³ for Web Applications fills this gap by providing the actual user experience and application end-to-end response time contribution detail. This enables IT personnel to “see” how their applications are performing as experienced by their real end-users and to correlate this experience to the underlying application components in terms of end-to-end performance contribution.

VERITAS i³ for Web Applications provides an integrated application performance focused solution that provides the actual end user experience and fills the end-to-end application performance-monitoring gap. Companies using the VERITAS i³ for Web Applications solution are able to proactively detect emerging friction points and isolate hotspots in minutes to avoid performance problems during all phases of the application life cycle. This empowers IT organizations to more effectively isolate the sources of problems in the reactive environment; not only by identifying those root issues but also ranking those that have the greatest impact and offering best practices correction recommendations. VERITAS i³ for Web Applications complements and seamlessly interfaces with your existing infrastructure framework.

VERITASeritas i³ for Web Applications uniquely provides an integrated application performance management solution using a combination of Web and application agents, end-to-end and top-down correlation technologies, information presentation technologies, a performance data warehouse, and problem isolation and correction assistance that complements your existing framework and infrastructure investments. This enables the proactive detection of potentially problematic application service level trends, identifies those with the highest pay-back and significantly reduces the time to proactively or reactively determine the hot spots, determine the impact of the performance anomaly, determine the cause of the performance anomaly as well as verify that the corrective action had the desired affect. Having VERITASeritas i³ for Web Applications in place to continually monitor and enable the improvement of your customer experience allows the organization to focus on the business associated with the web content. It helps organizations focus on application ROI rather than the delivery vehicle. In other words, VERITAS i³ for Web Applications connects the dots between performance information and how it relates to the achievement of the business goals.

THE VERITAS I³ METHODOLOGY
It is vital for all organizations to understand the importance of a systematic approach to coping with performance problems, and to realize the inherent ineffectiveness of ad-hoc fixes and problem solving. The foundation of the VERITAS i³ for Web Applications solution is a methodology designed to facilitate the rapid detection, isolation, analysis, corrective action and verification of application performance problems. All organizations can benefit from the discipline that VERITAS i³ software brings to the performance management process. The VERITAS performance management methodology enables you to use a proven process, as well as proven supporting solutions, to implement effective performance management within your organization.
The VERITAS performance management methodology comprises five stages:

- **Detect** – Proactively identify the symptoms that could indicate a performance problem.
- **Find** – Identify the problematic tier and application component: is it the network, Web server, application server or the database server.
- **Focus** – Discover the root cause of the problem.
- **Improve** – Take the steps required to improve performance.
- **Verify** – Make sure the steps taken have achieved the desired goal.

These stages combine to form a process that provides a systematic approach to finding and resolving all kinds of performance issues, both predictable and unforeseen.

For example the methodology begins with an automated proactive detection (DETECT) through baselines and SLAs, using the products alert capabilities. Alerts can be directed at the appropriate infrastructure framework, organization, or person based on what was detected. Alternatively, the appropriate person in your performance organization could periodically detect some long service times while reviewing the performance trends of elements such as long running URLs, COM+ components issuing long database calls, etc.

Depending on the issue detected the application manager can view the application, end-to-end, and isolate (FIND) the problematic tier and application component, which is source of the degradation. Problems could be caused for example because of network delays, specific to one remote location or to all locations, long running URLs, COM+/NET objects waiting for the response of long running SQL queries against the SQL Server database, etc. The process of finding the problematic component is end-user focused, so that we actually follow the end-user transaction or the interfaced application all the way, from the time activated by the user.

After finding the problematic component, an in-depth analysis of that component has to be done in order to focus on the root-cause of the problem (Focus). For example, finding that the problem happens because of a missing index in accessing one of SQL Server database table. Knowing the problematic SQL statement is not enough, we are also interested to know which COM+ object is initiating the problematic SQL statement and which application is impacted.

After focusing on the problem’s root cause comes the IMPROVE phase, fixing the problem. Continuing our previous example, adding an index through using Indepth for SQL Server recommendations fixes the problem.

Once the fix has been applied we should VERIFY that indeed we fixed the problem and it doesn’t reappear. Utilizing long-term information kept in the Performance Warehouse we can examine service time before and after the fix has been applied. Doing so shows whether we fixed the problem and service times are as expected or not.
VERITAS I³ FOR WEB APPLICATIONS

VERITAS I³ for Web Applications focuses on providing your organization with the ability to measure and monitor your application performance end-to-end from the browser, through the J2EE or COM+/.NET application server, optional additional middleware components and right into the SQL Server database and storage devices.

Conceptually VERITAS I³ software is comprised of VERITAS Insight, VERITAS Indepth and VERITAS Inform software functionality. Unlike frameworks the VERITAS I³ components can be purchased as separate products and run independently to meet specific environmental and application needs. The graphic below depicts the architectural components of VERITAS I³ software.

VERITAS Insight software functionality provides the agents and logic required to enable your IT organization to see the end-to-end response time contributions across the Client, Network, Web, application server, middleware servers and the SQL Server database servers and to isolate the components which are responsible for slowdowns and performance bottlenecks. The whole process starts by examining the end-user experience, measuring the actual Web-user response time at the application, transaction or page level. Then, upon detection of a response-time SLA or baseline breach, the problematic tier and component are found, as described earlier in the methodology section.
VERITAS Indepth™ software functionality provides in-depth drilldowns to identify the root cause of problems. It collects and correlates detailed component information. For example, J2EE detailed information, methods, SQL Server resource consumption breakdown, top slowest SQL statements, etc.

VERITAS Indepth for SQL Server provides detailed information on the application and the DB running on the SQL Server. We can get to the specific cause of the performance problem, monitor SQL Server users, programs and statements as well as getting suggestions on how to fix the problem.

One of the important features of VERITAS Indepth for SQL Server is its ability to link back problematic SQL statements to their initiating COM+ objects. This enables us to find and understand application components, which suffer from long database access times or execute heavy resource consuming transactions at the database level.

Another important feature of VERITAS Indepth for SQL Server is its ability to associate a logical database I/O with its physical device I/O for Windows file system running on different storage devices, including EMC, Hitachi and HP. This ability enables us to analyze I/O bottlenecks, for example two heavily database index files are accessed simultaneously and located on the same physical device – we detected a contention problem.

VERITAS Inform software functionality provides the real-time and near-time alerts that are driven off the baseline and thresholds defined. VERITAS Inform software includes a rich set of communication options and seamlessly interfaces to all the major infrastructure frameworks. VERITAS Inform functionality also includes the reporting components of i3. This rich report capability enables the presentation of ad hoc reports as well as scheduled reports. These reports are useful for service levels, trends and exception.

VERITAS i3 includes the VERITAS Performance Warehouse, which provides the common repository for long-term data history, collected by the different VERITAS agents along the application path: Web servers, application servers, middleware servers and database servers. This enables trend analysis, long-term capacity planning, baseline calculations based on historical data and correlated reports over time.

VERITAS i3 for Web Applications includes the functionality necessary to see the actual user’s response time, analyze the end-to-end path correlated contributors including analysis of Web server URLs, J2EE servlet and EJB response time and throughput or COM+ components accessing the SQL Server database, invoked SQL statements and database resource consumption breakdown.
VERITAS $i^3$ FOR WEB FEATURES

The StartPoint enables full end-to-end view of the application environment components, starting from the Web client, the Web server, application server and the SQL Server database. StartPoint provides a visual representation of the application performance of each of the application tiers as well as the actual application data flow between tiers.

StartPoint – Integrated view, from the Web client tier to the SQL Server database

StartPoint provides also an alert overview of the different components by categories, e.g. a service time problem at the Web tier or a performance problem at the SQL Server tier. This enables at a glance quick problem classification for each tier. The next step is to launch one of the $i^3$ components (Insight, Indepth, Inform - Alerts and Foresight) to start with.

A typical scenario could start with a user launching Insight to get an overall overview of his application environment or proactively receiving an alert indicating service level threshold has been exceeded. Long-term trend analysis and top resource consumers could be obtained using long-term metrics stored in the Performance Warehouse. Easy access to the PW information is done via Foresight, which generates the relevant reports.

PROACTIVE MANAGEMENT THROUGH EXCEPTIONS

Managing a large Web application environment with hundreds or thousands of users is difficult to impossible. The only way to do so is working proactively rather than reactively. Precise $i^3$ provides the way to do so through the
use of exceptions, based on SLAs and baselines. The graphics below shows an alert generated on the SQL Server tier.

Alerts could be generated on the end-user client tier breaching an SLA as well as on other tiers across the application infrastructure. Alerts are classified into categories such as performance alerts, load alerts, service alerts, etc. Clicking on an alert provides more information on the alert’s cause.

END-USER EXPERIENCE AND SLAS

End-users are the most important ones. If end-users are satisfied, application managers are satisfied. Precise i³ provides a way to measure response-times as exactly seen by end-users and to define SLA thresholds on those metrics. The graphics below shows an example for SLA breaches at remote locations and the times when breaches happened.
CORRELATED END-TO-END VIEW

Launching VERITAS Insight software provides a cross tier view of the different tiers and components. Activities could be examined by tier for a given time period. For example, we can look at the top N URLs invoked at the Web server tier or top N SQL statements at the SQL Server tier.

Using Insight SmarTune correlation, selecting one URL, for example the top URL, shows only activities in other tiers that are associated with the selected URL. This view type correlates activities across components, URL-to-SQL, which means you would be able to view for example the SQL statements executed by a specific URL.

The graphics below shows an example for an end-to-end overview.
On the upper left hand side, App Tiers – Total Time, we see the different tiers, Web server, J2EE application server and SQL Server in this example. We identify that the SQL Server tier is contributing the most to response time. Just below we can see the application App Tiers Data Flow, from the Web server to the J2EE tier and the SQL Server tier as well as directly from the Web server tier to the SQL Server tier.

The main part of the screen displays the main data entities for each component. For the Web server it is URLs, for J2EE tier it is invoked methods and for the SQL Server tier it is SQL statements. Examining the SQL Server part we can immediately identify a relatively long running SQL statements (the first statement).

IN-CONTEXT DRILL DOWN

VERITAS I³ enables in-context drill down. Selecting the top resource consuming SQL statement and launching Indepth for SQL Server opens Indepth for SQL Server at the right context, showing the selected statement in the same time frame with the relevant information necessary to analyze the problem. Doing this process manually is tedious and time-consuming. The graphics below shows how to launch Indepth for SQL server focusing on the slowest SQL statement.
EXPERT IN-DEPTH ANALYSIS

VERITAS Indepth products enable expert analysis of the monitored tier. In our example we drilled down in-context to the SQL Server database tier using Indepth for SQL Server and to the same SQL statement that was identified as the slowest one. In case of a J2EE application server, we could have drilled down to the J2EE server using Indepth for J2EE. Expert analysis provides very detailed performance data for the specific tier. For example we can view a detailed SQL Server resource consumption breakdown, SQL statements, users and programs. The graphics below presents a detailed resource consumption breakdown of the SQL statement under investigation. In addition to the resource breakdown, we can see that there was a change in resource consumption over time and the “Using CPU” time is growing over time.
When there is a performance problem, we would like to examine the Execution Plan. Indepth for SQL Server enables us to view the execution plan of a SQL statement. Understanding the access plan of SQL Server could make performance improvements to SQL statements. Looking at the execution plan we can see that an index was defined or changed improperly. The graphics below shows this.
Indepth for SQL Server has a strong unique feature of storing historical changes over time in a long-term repository, called the Performance Warehouse. If we examine changes over time, we can see that indeed the index was changed and this is what caused the problem. The graphics below shows this.
EXPERT RECOMMENDATIONS

A special feature recommends indexes for SQL statements. This feature is extremely valuable to novice DBAs as well as saves time to experienced ones. The picture below shows an example for recommended indexes.

COM+ CORRELATION

In many cases COM+ components are “black boxes” and we would like to get visibility and uncover the curtain behind them. We would like to understand which COM+ application, interface, component and method cause the problem, or impacted by the performance problem. The challenge is being able to correlate SQL statements and their origin COM+ methods that initiated these SQL statements. The motivation is to understand the application impact of problematic SQL statements and to identify which COM+ components access the database inefficiently.

A unique feature which exists in Indepth for SQL Server correlates a SQL statement back to its COM+ application, interface, component and method, providing the link to the source of the problematic SQL statement. It provides us information on the application function that executed the SQL statement. The graphics below shows such an example.
LONG-TERM ANALYSIS

Performance metrics collected from the various tiers (client tier, Web server, J2EE, COM+, SQL Server) are kept in a long-term repository called the Performance Warehouse. These performance metrics could be used for trend analysis and capacity planning. VERITAS I³ provides a mechanism, which utilizes the Performance Warehouse to generate hundreds of out-of-the-box smart reports. The graphics below shows a SQL statement resource consumption breakdown over time. We can see that using CPU time decreases sharply after adding the index.
METHODOLOGY
As mentioned earlier, VERITAS I³ software is based on a methodology that serves as guidance from problem detection to problem resolution: Detect, Find, Focus, Improve and Verify. We started from an alert, indicating response-times thresholds were breached; we looked at the end-to-end architecture, isolating the problem to SQL Server tier and found that an index problem. We also verified that after adding the index, response-time returned back to normal and we meet our SLA objectives.
BENEFITS OF VERITAS I³ FOR WEB APPLICATIONS

From a business perspective using VERITAS I³ for Web Applications can significantly reduce slowdowns, achieving fast ROI while maintaining low TCO.

VERITAS I³ for Web Applications help organizations to:

- Understand who is using the application, which applications are being used, when and from which location
- Proactively detect potential problems before it impacts your users
- Improve end-user satisfaction with the application
- Increase ROI - detect, find and solve problems quicker
- Reduce TCO – significantly reduce the amount of management time and number of people and organizations involved in the problem management process. Also less experienced people are required to solve a problem.
- Verify meeting of business level objectives
- Have one integrated correlated view of their application environment
- Conduct organization health checks through baselines and SLAs
- More effectively communicate the service level achievements and improvements to the organization
- Eliminate blamestorming between IT functionaries
- More effectively manage and tune the application to effectively utilize the system resources required to provide the mandatory service level.
- Use just-in-time capacity, deferring equipment and the associated software licenses and support costs until they are actually needed.
- Better understand future capacity requirements, based on current application usage and growth patterns


The solution can be extremely helpful in a development environment for systems testing and understanding performance. Additionally the comprehensiveness of the data captured, its correlation and the easy to focus on data presentation, as well as the automated analysis assistance and recommendations provided by Smartune can help application developers more effectively prioritize the focus. It can eliminate false starts and literally save hundreds of looking of “needles in the haystack when diagnosing problems discovered in QA and production. Using VERITAS I³ for Web Applications in QA, Staging and Production environments assures the developer more time to focus on application development and less time in support roles in problem resolution.

Similarly, the architecture, performance and capacity planning folks can significantly reduce the time associated with identifying opportunities to reduce service time and anticipate or proactively isolate performance problems. The immediate benefits include:

- Catch design and architectural issues early before they affect end-users
- Reduce the day-to-day tuning burden
- Quickly and automatically identify baselines and performance trends

All these capabilities being used throughout the application lifecycle can significantly reduce your company’s web initiative time to market while ensuring the achievement of the application ROI when in production by eliminating the performance frustrations that result in click-away, which directly affect customer retention.

In summary, VERITAS I³ for Web Applications is an excellent vehicle to measure and improve organization performance objectives, providing all the necessary mechanisms to realize significant savings in effort and
resources, while effectively managing your application performance proactively as well as being prepared to respond quickly and efficiently reactively.

**SUMMARY**

Web-based applications have become an essential component of every business’s delivery system in order to compete and realize the necessary savings and productivity demanded to achieve their bottom line objectives. Their deployment has dramatically changed the fundamental economics of service delivery, competitiveness and customer expectations and are replacing the more expensive and less responsive traditional “people assisted” services.

Experience has shown that there is a direct correlation between service level and productivity (internal) and revenue (external). There is also a direct correlation to expenses incurred to achieve the service level. The penalty for not meeting the users’ performance expectations may mean loss of business, loss of opportunity and failure to realize the ROI for the effort and operational expenses.

Delivering the right level of service has become very challenging as measuring the service levels and isolating performance hot spots in these new Web-based application systems is fundamentally more difficult due the inherent complexity introduced by multiple tiers, numerous old and new software technologies in the path and multi-node distributed computing, including messaging and transactions and database access.

**VERITAS i³ for Web Applications** addresses these challenges and actually simplifies the deployment and production management of high performance web-based applications. Using **VERITAS i³ for Web Applications** helps companies manage the service level, contain the costs, maximize the efficiency of the operational investment and improve the service levels experienced by the end users.

**VERITAS i³ for Web Applications** is the industry’s first software offering that provides a way to quickly, efficiently and unobtrusively capture the hard to get URL-to-SQL application metrics and present these in a manner that enables crisp communication, rapid proactive or reactive detection, correction and verification through-out the application life cycle.

**VERITAS i³ for Web Applications**, sets a new milestone in web application performance management by delivering a comprehensive integrated software solution that addresses the major IT challenges associated with the management off multi-tier Web application components.

This solution:

- Provides proactive management through exceptions, SLAs and baselines
- Provides a way to define and measure organizational business objectives and SLAs
- Meets the detection, reporting, isolation and diagnostic needs of all organizations involved
- Provides a view of the actual service levels as seen by all the real end-users at their desktop
- Provides an integrated correlated view – collects and correlates performance metrics from the different components along the application path to create an integrated correlated view
- Monitors 7x24 real users, real data and real Web transactions as opposed to simulated or synthetic activity monitoring
- Isolates the hotspots from the browser to the SQL and the database
- Isolates the problematic component: network, Web server, J2EE/COM+ application server or inefficient SQL Server access.
- Identifies root cause right down to the problematic SQL statement and COM+ object originated this statement
- Enables association between logical database I/O and its physical device I/O for Windows file system.
- Provides best practices corrective actions and recommendations
- Provides long term performance repository for analysis, trends and capacity planning
• Operates in a production environment with minimal overhead
• Fits all phases of the application lifecycle
• (Development, QA, Stress Testing, Staging and Production)
• Requires no application changes or developer awareness
• Interfaces with and complements the existing infrastructure frameworks