WHITE PAPER
IT Knowledge Exchange Series

MEASURING PRE-PRODUCTION APPLICATION, SYSTEM AND PERFORMANCE VOLUME STRESS TESTING WITH TEAMQUEST

A white paper on how to enhance your testing discipline

TeamQuest®

Contents

Summary 2
A Basic Testing Methodology 3
A Word about Testing Tools 3
The TeamQuest Approach 4
The Role of TeamQuest View 4
The Role of TeamQuest Model 5
Conclusion 5

Summary

These days just about everyone employs some level of application and system testing prior to introducing changes into the production environment. The goal of such testing is to ensure quality and minimize any possible disruption to business operations. In many cases in the past, application and operating system code quality was lacking, thus companies would incur unexpected production outages, mostly due to code or configuration parameter defects. More rigorous testing, along with more testing experience and discipline, has minimized the impact of code and parameter defects on production operations. As code and parameter defects are reduced, poor application performance is more frequently being seen impacting production operations. The problem most organizations face in trying to address performance problems is that building test platforms large enough to emulate production volumes is usually cost prohibitive. As a result, few companies perform performance related tests at actual production business levels. This paper will reveal how TeamQuest performance software can be employed as a more cost effective solution to identify potential performance problems during pre-production testing and reveal long term risks to capacity plans due to changes in resource consumption.

A Basic Testing Methodology

Testing best practices provide for a staged approach, with each ensuing stage containing a higher level of testing rigor and results analysis. The most common steps in an average testing procedure are:

- **Component testing – The first step. The application or system programmer applies changes or develops computer-related code (computer instructions) that satisfies the project specifications. Component testing is performed by the programmer to find any code defects and to ensure the program operates in accordance with the specifications. This work is generally performed in a small-scale (and in many cases shared) test server environment with minimum of testing rigor.**

- **Integrated or System testing – This step builds on the first and is designed to test interactions between the individual components within a project. The goal is to determine that functionality is maintained in an integrated operational environment and that each component, individually and as a whole, still operates according to project specifications. Integrated testing is usually performed in a stand-alone environment, although with limited testing volumes. The work is usually performed by the project's programmers.**

- **Quality Assurance (QA) testing – Once the programmers have completed their testing procedures, the code and related changes are turned over to a separate testing organization for independent verification. In the case of application releases, the testing team also ensures there are no conflicts between projects in a particular release, especially where shared programs (such as Microsoft Windows DLL files) are involved. Testing is performed using scripts so results can be repeated and impacts of changes understood. The test environment closely resembles actual production operations. In large organizations, maintaining this testing infrastructure can be quite costly.**

- **Performance Volume Stress (PVS) testing - A comprehensive test using actual production transaction volumes to validate proper operation of applications at expected business volumes and determine any adverse performance implications. Due to the amount of work involved to perform the tests, they are usually restricted to testing mission critical and high cost applications. Testing tools such as transaction script generators and load testers are usually employed to perform the tests, which are in most cases run by an independent group of testers. Since transaction loads are at production volumes, substantial infrastructure can be required to support testing. In large companies this can cost many millions of dollars to replicate production infrastructure.**

The testing process and procedures must also employ a continuous improvement facility. As a normal course of business, test procedures are refined over time as applications and infrastructure requirements change and additional implementation defects are discovered. Through continuous improvement processes, testing scripts and processes are modified to address the new discoveries, therefore substantially reducing or eliminating the risk of a reoccurrence.

An effective testing process ensures production changes are error-free and applications continue to perform at expected levels of service. You will know that your testing process is really effective when your end users and/or customers are totally unaware that anything in the IT environment, except for expected visual changes, has occurred.

A Word about Testing Tools

It is important for the testing process to be easily repeatable, for without repeatability there is no way the impact of a change can be accurately measured. In addition, most organizations have limited staffing resources available to support testing processes so tools increase staff productivity by automating manual testing and results recording processes.

There are a number of tools available in the marketplace as well as variety of open source and freeware solutions. Although some are integrated testing suites, the following categories of tools are commonly found in "best practices" testing labs and may be of value to an emerging testing organization.

- **Test Managers – these tools facilitate testing repeatability by identifying the steps in a particular test and tracking progress. As the steps are completed, the task is checked off and the subsequent task identified and pertinent action data made available to the tester. Depending upon the sophistication (and sometimes price) of a tool, testing process related forms are provided in paper format or accessed through a graphical interface on a personal computer.**

- **Scripting tools and Load Generators - are what the name implies, tools to script the steps of a test and apply simulated loads to test systems. Some tools capture actual data from performing a particular transaction or process and permit it to be replayed in the test environment. Many scripting tools also provide facility to scale number of transactions by generating additional test database and file records and adjust account and customer numbers to avoid disk hot spots. Doing so more accurately represents day-to-day production operations as a single customer rarely accounts for all the transactions in a single time period. Load generators take the scripts and simulate large numbers of terminals permitting server-based applications to be tested at higher volumes than capable by the limited testing staff.**

- **Scheduling tools –**
  Used to ensure computer jobs are completed in a particular sequence and that prerequisite data is available for processing. Schedulers also contain documentation on problem handling procedures should a particular job or process fail.

- **Comparators –**
  These tools employ automation techniques to compare test results against a previously defined baseline. Test and baseline data is processed by the tool and a "differences" report is generated.

The TeamQuest Approach

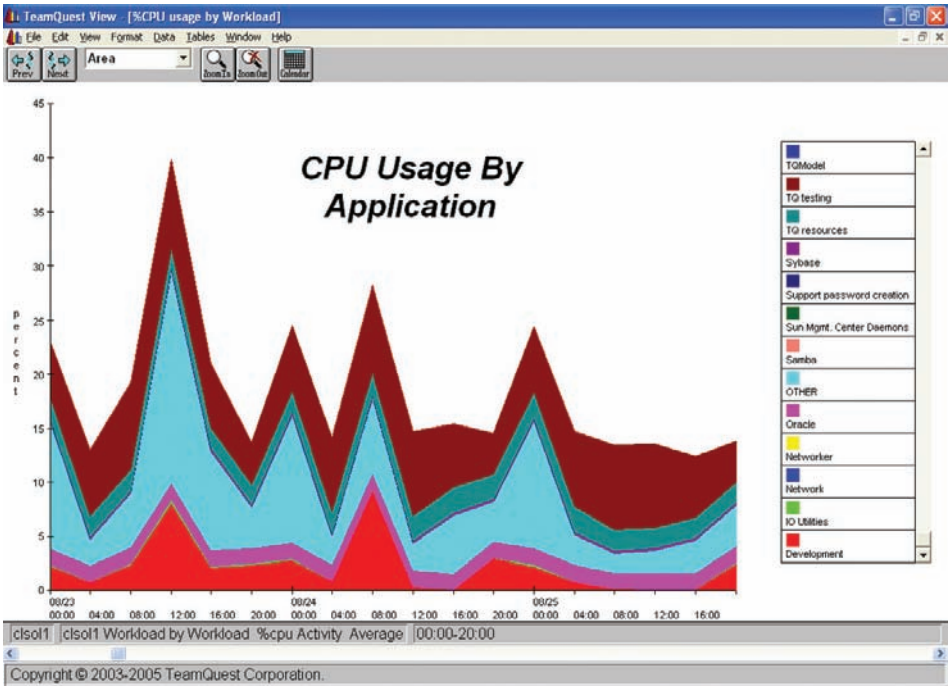
TeamQuest believes that testing should be more comprehensive than just finding code defects. The consequences of a poor performing application can actually have longer reaching service and financial impacts on an organization than the short term effects of a code or parameter defect. Therefore TeamQuest recommends a more balanced approach to testing by adding a performance evaluation methodology. In doing so, customers and end users can be assured that expected levels of service will continue to be maintained after change has been introduced into the production environment. Performance evaluation will also ensure that planned business usage of modified applications continues to track to capacity plans and where deviations occur, provide sufficient advance warning so corrective actions can be completed before performance becomes a problem.

The Role of TeamQuest View

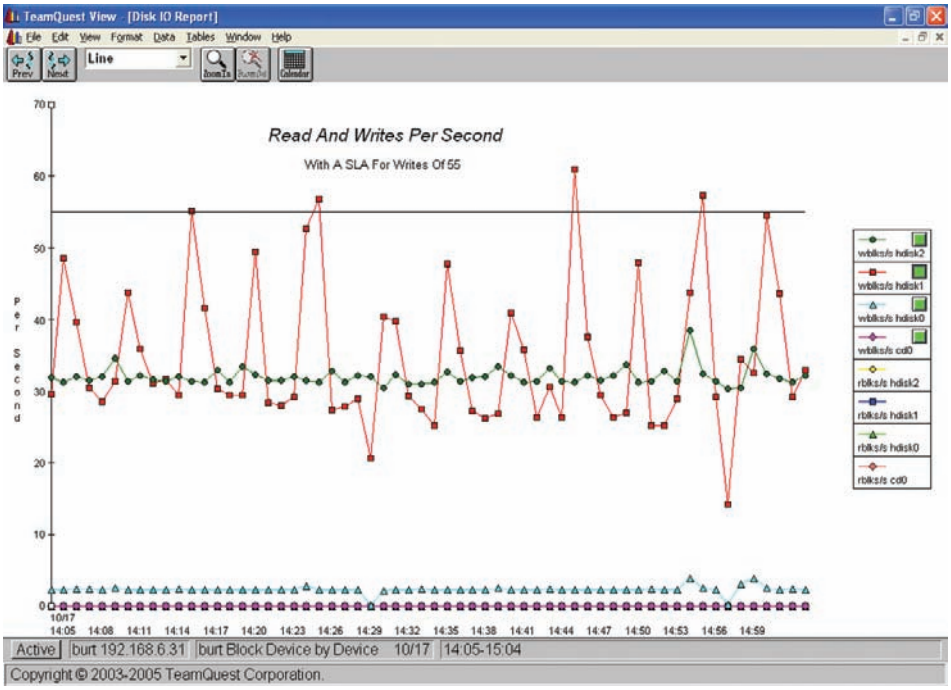
TeamQuest View is a multi-platform performance management tool. Agents gather performance data from the processes running on the platform and store it in a historical database. Data can be archived for future reference. Since baseline measurements are necessary to compare test results, TeamQuest View provides an excellent means of storing baseline and testing performance data. View has the flexibility to analyze a single application on a single server or a single multi-tier application across a number of diverse servers or any combination in between. Built-in reporting facility provides excellent method of comparing results.

At a minimum, the following performance data should be analyzed to determine the performance and capacity impacts of implementing a change:

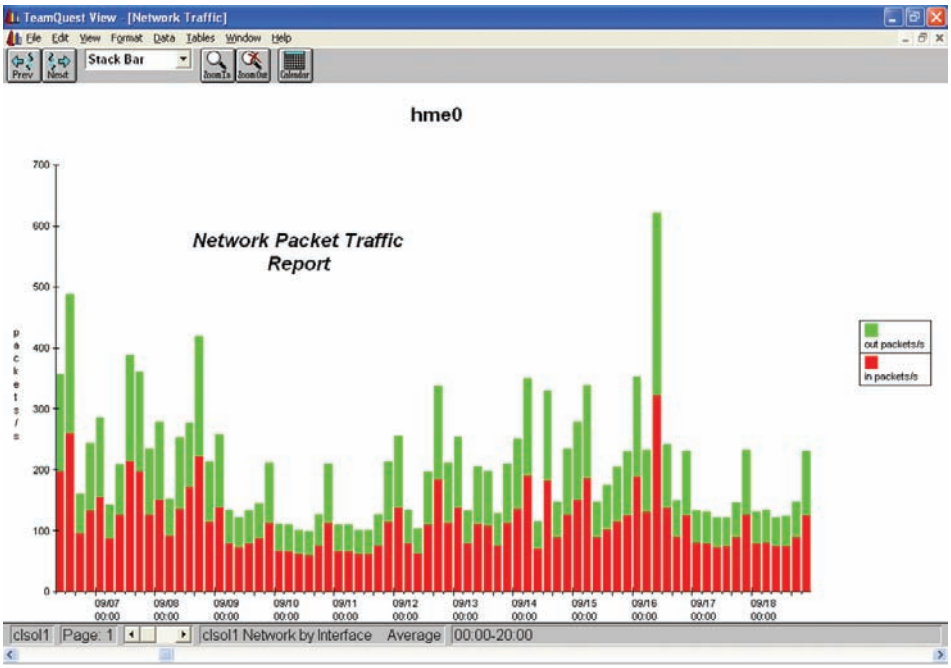
- **CPU consumption –**
  the amount of resources a particular transaction or process consumes. This analysis can also validate that multi-processing capabilities have been preserved after the change.



- **I/O performance –**
  validates that number of disk reads and writes fall within specifications and that changes have not introduce abnormal volumes that would impact application performance



- **Network performance –**
  at a minimum ensures that number of network data packets have not substantially increased as a result of the changes. Increased network traffic between the servers and offices can cause network bandwidth capacity, dedicated lines and/or CIRs, to be exceeded, thus incurring additional, possibly unplanned charges.



Using TeamQuest in conjunction with other testing tools provides a very effective, cost effective way to understand the immediate impacts of application and system changes on server consumption. The historical database provides excellent way to maintain a history of application performance as an application is enhanced and can provide simple way to validate the financial assumptions associated with a project's Total Cost of Ownership (TCO) and Return On Investment (ROI) documents. The history can also provide the necessary data for lessons learned analysis and used to improve the application programming process.

The Role of TeamQuest Model

TeamQuest Model is an analytical capacity modeling tool. It employs the use of mathematical formulas to simulate growth based upon baseline performance measurements. Model has an easy to use interface to extract performance data from the View historical database. Where View is a quick and simple way to compare changes in application infrastructure consumption as a result of changes, Model, although easy to use, takes more time to set up and perform the work. TeamQuest recommends starting with mission critical, high visibility and/or high infrastructure resource consuming applications, then addressing applications of lesser business importance as time and staffing resources permit.

TeamQuest Model used separately or in conjunction with other testing tools can provide substantial value to the organization in the following ways:

- **Cost avoidance. Model capabilities permit real world production business volumes to be simulated using smaller test platforms, thus eliminating the need for expensive replications of production equipment and software. In addition, Model can simulate operations on different hardware platforms, therefore testing can be performed on less costly equipment and production work volumes simulated on the higher capacity production equipment with high degree of accuracy.**
- **Capacity Plan Validation. The results of application testing can be introduced into existing infrastructure capacity models. Previous assumptions can be validated and if anomalies exist, plans can be adjusted and corrective actions, if needed, can be taken well before performance problems are experienced by customers and end users.**
- **Application Sourcing Analysis. Model capabilities permit simulation of an application on different computing platforms, therefore application performance can be analyzed across a variety of hardware configurations. For example the testing team can analyze how an application performs on a platform with fewer, faster processors as well as in a multi-platform high availability cluster environment. All can be accomplished without needing physical hardware in place to perform the testing. The best solution satisfying an application's performance and capacity needs can easily be determined in advance with high degree of accuracy.**

Conclusion

TeamQuest View can provide performance monitoring and historical comparisons, permitting early identification of performance problems. TeamQuest Model can simulate production work volumes, reducing testing infrastructure requirements. Together they provide a simple, easy-to-use, and cost effective supplement to testing tools, permitting a more comprehensive approach to the application testing process.

If you like what you have read, click here for a

FREE SUBSCRIPTION

to the TeamQuest Business Intelligence Series.

TeamQuest Corporation

www.teamquest.com

Americas
One TeamQuest Way
Clear Lake, Iowa 50428
USA
+1 641 357-2700
+1 800 551-8326
info@teamquest.com

Europe, Middle East and Africa
Box 1125
405 23 Gothenburg
Sweden
+46 (0)31 80 95 00
United Kingdom
+44 1562 881 889
emea@teamquest.com

Asia Pacific
Level 6, 170 Queen Street
Melbourne, VIC 3000
Australia
+61 3 9641 2288
asiapacific@teamquest.com

Copyright © 2005 TeamQuest Corporation
All Rights Reserved

TeamQuest and the TeamQuest logo are registered trademarks in the US, EU, and elsewhere. All other trademarks and service marks are the property of their respective owners. No use of a third-party mark is to be construed to mean such mark's owner endorses TeamQuest products or services.
The names, places and/or events used in this publication are purely fictitious and are not intended to correspond to any real individual, group, company or event. Any similarity or likeness to any real individual, company or event is purely coincidental and unintentional.
NO WARRANTIES OF ANY NATURE ARE EXTENDED BY THE DOCUMENT. Any product and related material disclosed herein are only furnished pursuant and subject to the terms and conditions of a license agreement. The only warranties made, remedies given, and liability accepted by TeamQuest, if any, with respect to the products described in this document are set forth in such license agreement. TeamQuest cannot accept any financial or other responsibility that may be the result of your use of the information in this document or software material, including direct, indirect, special, or consequential damages.
You should be very careful to ensure that the use of this information and/or software material complies with the laws, rules, and regulations of the jurisdictions with respect to which it is used.
The information contained herein is subject to change without notice. Revisions may be issued to advise of such changes and/or additions.
U.S. Government Rights. All documents, product and related material provided to the U.S. Government are provided and delivered subject to the commercial license rights and restrictions described in the governing license agreement. All rights not expressly granted therein are reserved.