




TUNING MICROSOFT SERVER CLUSTERS

Guaranteeing High Availability
for Business Networks

- 
- Proven methodology to boost performance and availability
 - Step-by-step system optimization
 - Identify bottlenecks
 - Calculate capacity needs

ROBERT W. BUCHANAN, JR.

Tuning Microsoft Server Clusters

Guaranteeing High Availability for Business Networks

Robert W. Buchanan, Jr.

McGraw-Hill

**New York Chicago San Francisco Lisbon London Madrid
Mexico City Milan New Delhi San Juan Seoul
Singapore Sydney Toronto**

Cataloging-in-Publication Data is on file with the Library of Congress

Copyright © 2004 by The McGraw-Hill Companies, Inc. All rights reserved.
Printed in the United States of America. Except as permitted under the
United States Copyright Act of 1976, no part of this publication may be
reproduced or distributed in any form or by any means, or stored in a
database or retrieval system, without the prior written permission of
the publisher.

1 2 3 4 5 6 7 8 9 0 DOC/DOC 0 9 8 7 6 5 4 3

ISBN 0-07-141739-7

*The sponsoring editor for this book was Judy Bass and the production
supervisor was Pamela A. Pelton. It was set in Century Schoolbook by
Westchester Book Services. The art director for the cover was Handel Low.
Illustrations by: Aesthetic License / Kristine Buchanan.*

Printed and bound by RR Donnelley.

McGraw-Hill books are available at special quantity discounts to use as
premiums and sales promotions, or for use in corporate training programs.
For more information, please write to the Director of Special Sales,
McGraw-Hill Professional, Two Penn Plaza, New York, NY 10121-2298.
Or contact your local bookstore.



This book is printed on recycled, acid-free paper containing a
minimum of 50% recycled, de-inked fiber.

Information contained in this work has been obtained by The McGraw-Hill
Companies, Inc. ("McGraw-Hill") from sources believed to be reliable. However,
neither McGraw-Hill nor its authors guarantee the accuracy or completeness
of any information published herein and neither McGraw-Hill nor its authors
shall be responsible for any errors, omissions, or damages arising out of use
of this information. This work is published with the understanding that
McGraw-Hill and its authors are supplying information but are not attempting
to render engineering or other professional services. If such services are
required, the assistance of an appropriate professional should be sought.

This book is in memory of Kodi, the best friend and companion anyone could ever ask for, even if her only words were “Woof! Woof!”

Plus, this book is dedicated to all the people who have pioneered our current technology age, and more specifically, to all those who each day keep it up and running, often through great personal dedication and a continual state of exhaustion.

Preface

In today's business environment where there is financial uncertainty, shrinking IT budgets, and the need to reduce total cost of ownership and maximize return on investment, it is essential to have a proven methodology for determining, rather than guessing, how to resolve system bottlenecks, increase performance/throughput, and plan for increased capacity *at the lowest cost*.

While a single server is difficult enough to analyze, multi-tiered server clusters and farms are often daunting for even the most technical support groups. In these complex topologies, not only must each server and application be analyzed, but also the interactions and loads between them, including load balancing, data replication, and fail-over complexities, must be understood for a complete picture of the system. And when problematic performance problems are encountered, an understanding of the end-to-end system and infrastructure is usually required to analyze user response-time degradation and availability problems.

If you break the need to understand your system into smaller, more digestible bites, ask yourself:

- *Can I pinpoint the problems that are causing user response time or availability complaints?*
- *Can I isolate a bottleneck within my server cluster with certainty, or is it mostly guesswork? And, once isolated, do I know what is required to fix it?*
- *Do I know if threshold alerts really indicate a problem or reduced performance/response time? Or, are they just arbitrarily set levels that we spend a lot of time measuring and worrying about?*
- *Do I know if I have the correct amount and configuration for each critical resource (number of servers, infrastructure links, CPU, disk, memory, application design and settings, and many others) for best overall system performance?*
- *Do I know how much system resource each incremental user requires so I can accurately plan for scalability and capacity?*
- *Or, could I achieve the required capacity by simply re-deploying my hardware and applications within the cluster?*

If some or all of these questions hit home, then please read this book. *But one of my motivations in writing this book is also to deliver a broader message that proactive systems analysis is less expensive and more beneficial than reactive monitoring and fire fighting.* And, if you already believe this, but don't know how to proceed, this book provides you a proven road map for proactive, end-to-end systems analysis.

Proactive systems analysis requires a structured methodology that is capable of persistently relating Load to Resource Utilization to Response throughout the system so that you can

- *Resolve or mitigate problems before they occur.*
- *Determine the optimal configuration for your system load heuristics.*
- *Accurately determine additional resource needs to meet capacity and scalability requirements.*

There are many systems management and testing products on the market today that are very effective for collecting measurements and generating applied loads on a system. However, they fall short of providing in-depth analysis capability. In fact, they often create data overload. There is so much data generated that it is impossible to sort through it to determine what is important and what is not. And, it is often not synchronized sufficiently to allow the detection and isolation of system constraints and resource problems. You can see the symptom, but NOT isolate the cause.

The methodology described in this book explains how to configure instrumentation and collect critical time-persistent measurements using products you probably already have deployed, then how to utilize this data for a level of in-depth analysis of the system you have not had access to in the past. The methodology has been proven through hundreds of projects encompassing over 30,000 hours. It is

- Well structured.
- Adept at using proven techniques from other disciplines, such as constrained optimization based on finite element modeling, stimulus-response, relational trending, and critical pathing, all of which are explained in this book.
- Topology, system and application agnostic.
- Capable of being used for a single server, for a server cluster or farm, or for an end-to-end system analysis.
- Relevant for homogeneous or heterogeneous system configurations.

The initial chapters take you through the step-by-step process of our AR-Analysis methodology. AR relates the following three key system interactions:

- Applied load on the system *versus*

- Resource utilization of various system components *versus*
- Response to the applied load

The methodology includes system baselining, configuring measurements and data collection, analyzing the measurements and drawing conclusions or “actionable recommendations,” as we call them. Actionable recommendations are as they imply, specific changes, such as reconfiguring a server, adding more memory, redistributing applications, changing application settings, or whatever is appropriate to resolving the anomalies detected in the analysis.

The last few chapters provide real-world case studies of two projects in which the methodology was employed. The case studies outline the process steps and discuss the results and conclusions of the analysis. These chapters will help cement in your mind how the methodology is utilized across two complex server clusters.

I believe that you will find this process very beneficial and cost-effective for maximizing your system’s capabilities and providing high-quality service to your customer base.

While there are many books on the market that discuss installing and configuring Microsoft operating systems and applications, I think you will find this a unique perspective on how to analyze and optimize the systems. As discussed in the book and shown in the Chapter 8 case study, the process is readily adaptable to any application or operating system.

And, to assist you in applying the methodology and analysis techniques described herein, visit my Web site at www.robertwbuchanan.com or e-mail me at robertwbuchanan@mindspring.com to learn about the AR-Analysis toolkit. The toolkit includes Excel worksheets for many of the more critical and complex computations and example templates for Microsoft PerfMon measurement collection. It also includes worksheets for consolidating and trending elemental measurements and much more that you will learn about in this book. It is available for a nominal fee.

Acknowledgments

I would like to thank my wife, Kris, who created all the exceptional graphics for this book and for all my previous books. We work closely together each and every day and have for many years. And, we still really like each other, which I am told by many people is our greatest accomplishment.

And, I can't forget my other best friends, Kodi and Shiloh, two of the largest dogs you'll ever meet. Kodi was with us when I started the book, but never saw it finished. Instead we now have a new cohort, Kamots. These friends don't help me write, edit, or review the book material, but they sure help me relax after a day of writing.

Contents

Preface	ix
Acknowledgments	xiii

Chapter 1. The Cardinal Rule	1
Chapter 2. Methodology	39
Chapter 3. Instrumentation and Measurement	77
Chapter 4. Baselineing and Analysis	141
Chapter 5. The AR-Analysis Methodology	175
Chapter 6. Stimulus-Response	225
Chapter 7. Case Study—Server Cluster Analysis	239
Chapter 8. Case Study—Philip Services Corporation (PSC)	267
Chapter 9. Bringing It All Together	307

Index	313
-------	-----

The Cardinal Rule

Cause and Effect

Isaac Newton's Third Law of Motion states that "For every action there is an equal and opposite reaction." This law relates to the physical interaction that can be visually perceived between two different objects by Newtonian physics. For example, when a batter hits a baseball, the energy imparted to the ball makes it fly through the air. However, the bat and batter also have a measurable and equal load or reaction imparted to them caused by the ball's impact with the bat, as shown in Figure 1.1. This is why sometimes the batter's hands "sting" after hitting the ball. Today, Newton's law is a well accepted physical fact that has been proven uncounted times over the last three centuries.

Over 30,000 hours of systems analysis and product testing by myself and my associates has empirically demonstrated that computer systems, composed of components such as servers, clients, the network infrastructure, operating systems, and applications, have a corollary to Newton's theory. This corollary can be stated as, "For every load imposed on the system, some measurable amount of each component's resource is utilized to respond to the imposed load."

This means that there is a definitive and measurable relationship, which we call the AR-relationship, between the

- Applied load on the system
- Resource utilization of various system components
- Response to the applied load

While this is a relatively easy assertion to make, because of the complexity of most computer installations and systems, it is a difficult relationship to measure. Therefore, today's monitoring and analysis tools don't attempt to persistently measure or relate these three critical interactions. Rather, these tools collect and consolidate large amounts of data, which obscures the AR-

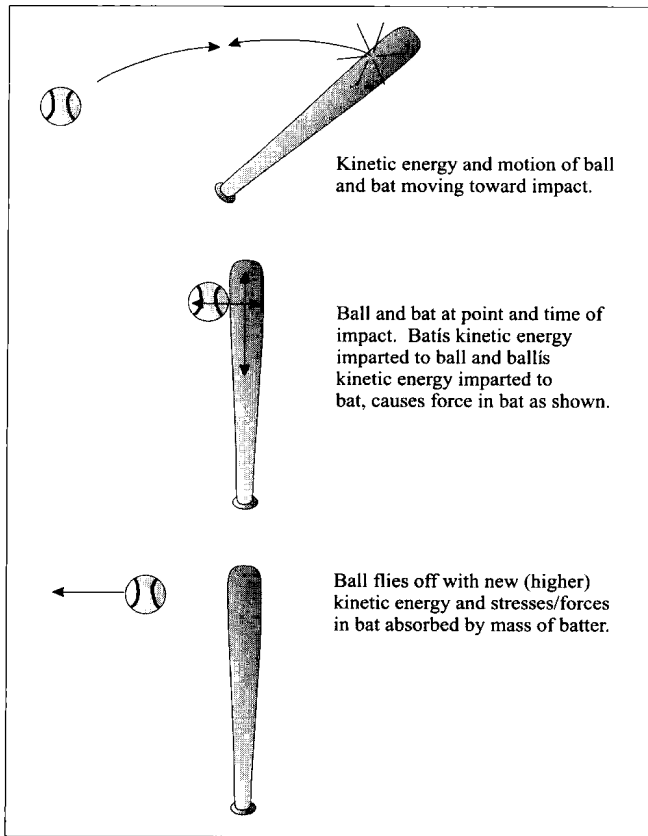


Figure 1.1 For every action there is an equal and opposite reaction.

relationships and inhibits efforts to determine the root-cause of problems relating to anomalies caused by these interactions. Only root-cause analysis can point the way to alleviating problems such as bottlenecks, poor response, capacity, and scalability. And, accurate root-cause analysis requires detailed measurements of the AR-relationship parameters.

Consider a typical networked system and applications, as shown in Figure 1.2. For such a configuration, system-monitoring tools are often deployed through a centralized SNMP console to collect and track key component resources, such as

- Network percent utilization
- Network errors
- Server CPU percent utilization
- Server disk percent utilization

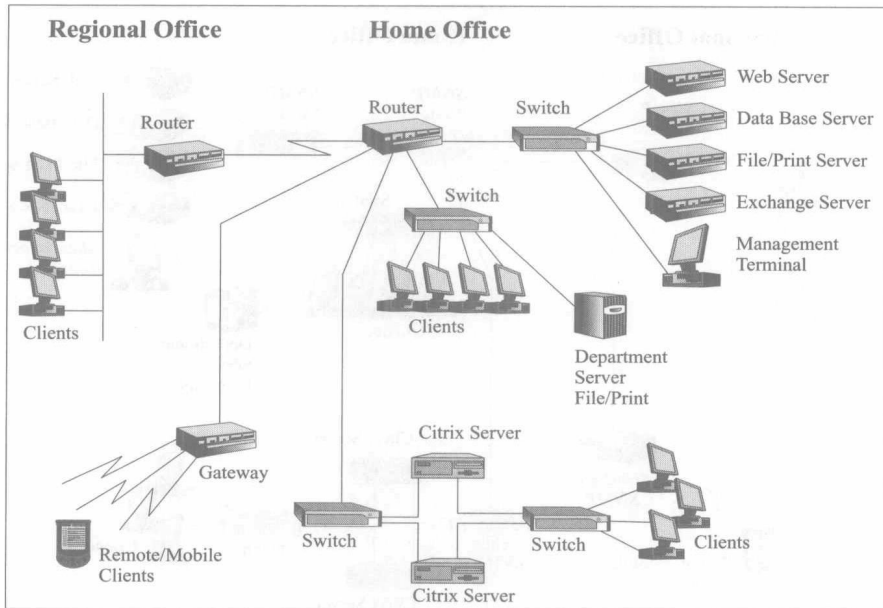


Figure 1.2 Typical networked system and applications.

- Server network interface percent utilization
- Server memory percent utilization

And through an application specific monitor, or exposed application measurements, other data can be collected. For example,

- Citrix MetaFrame XP parameters, such as
 - Number of users
 - Number of threads
 - CPU utilization
- SQL database parameters, such as
 - User connections
 - Cache hit ratio for prepared SQL plans
 - Cache hit ratio for ad-hoc SQL plans
 - Lock waits/second
 - Lock timeouts/second

Based on specified thresholds for each measurement, system alerts can be triggered as thresholds are exceeded, and historical trends of component resource utilization can be collected and tracked.

The instrumented system, shown in Figure 1.3, is typical of many systems today, in which problematic symptoms are tracked, but the underlying prob-

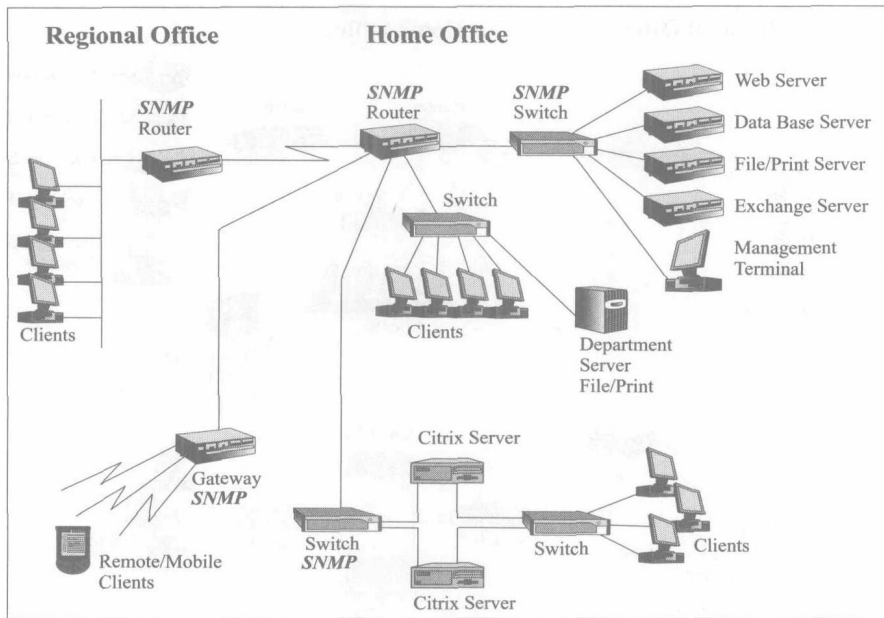


Figure 1.3 Instrumented networked system and applications.

lem cause is not determinable directly from the system measurements collected. This is because problems that exist cannot be resolved simply through correlating consolidated resource measurements, thresholds, and alerts.

Let's consider Figure 1.4, which shows collected measurements over several time periods. The SNMP network measurements are recorded for 5-minute intervals. The server SNMP measurements are recorded for 1-minute intervals, the Citrix parameters are updated in roughly real-time (3 seconds), and the SQL measurements are recorded every 30 seconds. Therefore, when the Citrix parameter for number of threads shows an alert condition at time stamps 14:44:49 and 14:44:52, there is no possible way to tell if system resources are becoming constrained or if user response time is degrading. This is because the Citrix alert occurred between the server measurements and at 2 minutes into the 5-minute network measurement interval. By the time the next measures on the server and network are recorded, any details pertinent to the Citrix event will be obliterated by the consolidation of the data over the extended time periods.

While this instrumentation can provide data on which to base alerts and historical trends, it is not suited to root-cause analysis. And, as we'll discuss shortly, it is not even very useful in helping to establish a realistic threshold on which to base an alert.

Many of today's tools, however, though rarely used, do provide the capability to collect discrete resource measurements, and by using them as part of an

[illegible]

Figure 1.4 (Continued)

[illegible]

overall analysis methodology the above AR-relationships can be determined. So why don't IT organizations do this today? Because most don't know how! They don't have experience on how to collect the required data, or how to analyze the data. These organizations are usually very good at monitoring systems, responding to alerts, fighting fires, and deploying new systems, but don't have experience with methodologies for data collection and root-cause analysis of complex computer systems and networks.

For example, if collection times are too large then detail will be lost and the data analysis cannot detect problem cause, as noted above. If times are too small or if the system is large, as many are, voluminous data will be collected, and its sheer volume will make it too daunting a task to properly analyze. And, since adequate analysis tools do not exist today, the process requires extensive technical effort by a very qualified individual to isolate problems and root-causes.

So, what is needed?

What is needed is a proactive analysis methodology that can provide actionable recommendations for resolving many of today's system issues for a reasonable investment in time and effort. The methodology must include instrumentation, measurement, and analysis processes, such as we have evolved over many projects that are effective, efficient, and employable across many different end-to-end system topologies and configurations. Required tool features exist today, although not typically deployed properly across the system, to collect time persistent component measurements. And the methodology and procedures described in this book outline how to use proven analysis techniques to provide actionable recommendations for removing bottlenecks, increasing capacity, and improving overall system effectiveness.

What Is Proactive Analysis?

Proactive is a term used and often abused by many vendors and consultants. Most products that tout proactive analysis are actually more reactive than proactive, detecting post-event symptoms, but not actually analyzing problem cause.

Proactive system and server-cluster optimization, on the other hand, actively measures and analyzes system resource utilization with the objective of detecting existing constraints before they result in serious problems. It provides a proven approach for improving system and application performance, availability, and capacity, by addressing problems before they occur (and often before their symptoms are even noticed by the users) and cause serious system problems.

As discussed in this book, server cluster optimization and tuning incorporates bottleneck detection, tuning, and capacity prediction within the server cluster and its supported applications and data stores, as shown in Figure 1.5. And, because server clusters don't exist in a vacuum, it also encompasses end-