

Content Server - Performance Tuning Guide
10g Release 3 (10.1.3.3.0)

March 2007

Content Server - Performance Tuning Guide, 10g Release 3 (10.1.3.3.0)

Copyright © 2007, Oracle. All rights reserved.

Contributing Authors: Jean Wilson

The Programs (which include both the software and documentation) contain proprietary information; they are provided under a license agreement containing restrictions on use and disclosure and are also protected by copyright, patent, and other intellectual and industrial property laws. Reverse engineering, disassembly, or decompilation of the Programs, except to the extent required to obtain interoperability with other independently created software or as specified by law, is prohibited.

The information contained in this document is subject to change without notice. If you find any problems in the documentation, please report them to us in writing. This document is not warranted to be error-free. Except as may be expressly permitted in your license agreement for these Programs, no part of these Programs may be reproduced or transmitted in any form or by any means, electronic or mechanical, for any purpose.

If the Programs are delivered to the United States Government or anyone licensing or using the Programs on behalf of the United States Government, the following notice is applicable:

U.S. GOVERNMENT RIGHTS Programs, software, databases, and related documentation and technical data delivered to U.S. Government customers are "commercial computer software" or "commercial technical data" pursuant to the applicable Federal Acquisition Regulation and agency-specific supplemental regulations. As such, use, duplication, disclosure, modification, and adaptation of the Programs, including documentation and technical data, shall be subject to the licensing restrictions set forth in the applicable Oracle license agreement, and, to the extent applicable, the additional rights set forth in FAR 52.227-19, Commercial Computer Software--Restricted Rights (June 1987). Oracle USA, Inc., 500 Oracle Parkway, Redwood City, CA 94065.

The Programs are not intended for use in any nuclear, aviation, mass transit, medical, or other inherently dangerous applications. It shall be the licensee's responsibility to take all appropriate fail-safe, backup, redundancy and other measures to ensure the safe use of such applications if the Programs are used for such purposes, and we disclaim liability for any damages caused by such use of the Programs.

Oracle, JD Edwards, PeopleSoft, and Siebel are registered trademarks of Oracle Corporation and/or its affiliates. Other names may be trademarks of their respective owners.

The Programs may provide links to Web sites and access to content, products, and services from third parties. Oracle is not responsible for the availability of, or any content provided on, third-party Web sites. You bear all risks associated with the use of such content. If you choose to purchase any products or services from a third party, the relationship is directly between you and the third party. Oracle is not responsible for: (a) the quality of third-party products or services; or (b) fulfilling any of the terms of the agreement with the third party, including delivery of products or services and warranty obligations related to purchased products or services. Oracle is not responsible for any loss or damage of any sort that you may incur from dealing with any third party.

Table of Contents



Chapter 1: Introduction

Overview	1-1
About This Guide	1-1
Related Documentation	1-2
Audience	1-2
Conventions	1-3

Chapter 2: Introduction to Optimization

Overview	2-1
Finding System Slowdowns	2-1
Typical Causes of Slowdowns	2-2
System Hardware	2-2
System Configurations	2-2
Other Questions to Ask	2-3
Steps in the Tuning Process	2-4

Chapter 3: Benchmarking

Overview	3-1
Designing Your Benchmark	3-1
Benchmark Design Points	3-2
Review and Document Your Current System	3-2
Mimic a Typical Session	3-3
Benchmark Tools	3-4
Running the Benchmark	3-6
Analyzing the Results	3-7
Search and Index Problem Areas	3-7
Database Problem Areas	3-8

- Network Problem Areas 3-9
- Core Engine/JVM Problem Areas 3-9
- File System/Memory Problem Areas 3-10
- Web Security Filter Problem Areas 3-10
- Analysis Tips 3-11

Chapter 4: Tuning the Core Content Server

- Overview 4-1
- Adjusting Configuration Flags 4-1
- Index Tuning 4-2
 - Working with the Search Index 4-3
 - About Full-Text Indexing 4-4
 - Indexing with Add-On Integrations 4-4
 - Database Full-Text Indexing 4-4
- Idoc Script Caching 4-5
- Database Tuning 4-5
 - General Database Maintenance 4-6
 - Backup Planning 4-7
 - System Maintenance 4-7
 - Other Database Maintenance 4-7
- Java Tuning 4-8
- Other Tuning Options 4-9

Chapter 5: Tuning Add-On Components

- Overview 5-1
- Batchloader 5-1
- Dynamic Converter 5-2
- Site Studio 5-2
- Publishing Utility/Connection Server 5-3
- Inbound Refinery 5-4
- Content Publisher 5-4
- Content Integration/Portal Suite 5-5
- Tools to Assist Optimization 5-5
 - Content Tracker 5-5
 - Compression 5-5

Chapter 6: System Architecture Tuning

Overview6-1

General Tips for Performance Improvements6-1

 Alleviating Storage Problems.....6-3

Common Mistakes6-5

Index

INTRODUCTION

OVERVIEW

This chapter provides an overview of performance tuning and the conventions used in the book. It contains the following topics:

- ❖ [About This Guide](#) (page 1-1)
- ❖ [Audience](#) (page 1-2)
- ❖ [Conventions](#) (page 1-3)

ABOUT THIS GUIDE

This guide provides an overview of techniques you can use to optimize the performance of Content Server. Like troubleshooting, the performance tuning procedure benefits from a systematic approach with a clearly defined set of steps to achieve your goal. However unlike troubleshooting, there often isn't a clearly defined problem area that needs to be addressed. This document will help you determine the best ways to approach tuning in order to achieve the maximum benefits.

There are three main types of tuning that can be done:

- ❖ **External system tuning:** This includes checking the memory usage, firewalls, or caching on your networks.
- ❖ **Content Server tuning:** This includes changing default parameters and software settings that affect the core Content Server performance.

- ❖ Content Server adjunct tuning: This includes tuning those things which the Content Server uses as resources, such as add-on components, databases, and indexes.

Some tuning and benchmarking is dependent on the use of the Content Server, whether it be primarily used for consumption or contribution. This document focuses on systems used primarily for consumption.

This document has the following chapters:

- ❖ [Chapter 1 \(Introduction\)](#), which describes the document structure and support options.
- ❖ [Chapter 2 \(Introduction to Optimization\)](#), which describes specific steps in creating an optimization plan.
- ❖ [Chapter 3 \(Benchmarking\)](#), which describes how to create and use benchmarks to determine your optimization progress.
- ❖ [Chapter 4 \(Tuning the Core Content Server\)](#), which describes changes you can make to the content server, index, and database to change its performance.
- ❖ [Chapter 5 \(Tuning Add-On Components\)](#), which describes tuning possibilities of different add-ons that can be used with Content Server.
- ❖ [Chapter 6 \(System Architecture Tuning\)](#), which describes how to tune different aspects of your system architecture to improve performance.

RELATED DOCUMENTATION

The following manuals may also assist you in optimizing system performance:

- ❖ *Troubleshooting Guide*
- ❖ *Managing System Settings and Processes Guide*
- ❖ *Planning and Implementation Guide*
- ❖ *Choosing a Search Solution*
- ❖ *Reverse Proxy Server Resource Guide*





AUDIENCE

This guide is intended for system developers and administrators who are trying to achieve faster performance of Content Server and its products.

CONVENTIONS

The following conventions are used throughout this guide:

- ❖ The notation `<Install_Dir>/` is used to refer to the location on your system where the content server instance is installed.
- ❖ Forward slashes (`/`) are used to separate the directory levels in a path name. A forward slash will always appear after the end of a directory name.
- ❖ Notes, technical tips, important notices, and cautions use these conventions:

Symbols	Description
	This is a note. It is used to bring special attention to information.
	This is a technical tip. It is used to identify information that can be used to make your tasks easier.
	This is an important notice. It is used to identify a required step or required information.
	This is a caution. It is used to identify information that might cause loss of data or serious system problems.

INTRODUCTION TO OPTIMIZATION

OVERVIEW

Performance tuning, often called optimization, consists of defining areas of your system that need improvement then defining an approach to tuning. Optimization is an iterative process and varies widely depending on architecture, infrastructure, and usage.

This chapter covers the following optimization topics:

- ❖ [Finding System Slowdowns](#) (page 2-1)
- ❖ [Typical Causes of Slowdowns](#) (page 2-2)
- ❖ [Other Questions to Ask](#) (page 2-3)
- ❖ [Steps in the Tuning Process](#) (page 2-4)

FINDING SYSTEM SLOWDOWNS

In general, you will see obvious signs of slow performance when performing the following Content Server tasks:

- ❖ searching
- ❖ batchloading or contribution, particularly when importing large chunks of content
- ❖ page rendering
- ❖ publishing, or ‘pushing’ the content elsewhere

This document will help you identify what constitutes slow performance and help you find where these slowdowns occur.

Tuning the Content Server follows the same basic rules as tuning any web application:

- ❖ Adjust generic configurations for your specific needs
- ❖ Optimize your third-party software
- ❖ Maximize your hardware use
- ❖ Optimize your cache use

These topics are discussed in more detail later in this document.

TYPICAL CAUSES OF SLOWDOWNS

There are usually two general areas that can slow your Content Server performance: inadequate system hardware or an inadequate system configuration. After you have verified that your hardware and configuration are not the source of your performance problems, then you can move on to examine Content Server tuning.

System Hardware

A good place to start performance evaluation is with system hardware. If system performance is poor, check the following areas to make sure they are adequate for the task:

- ❖ File system size: If your file system is inadequate or too slow, search performance can be slowed. Make certain that you have adequate space for the number of servers your site is using.
- ❖ Memory size: If your memory size is too small, swap times can slow the system considerably. Make certain that you have enough memory and that it is not over-used or under-used.
- ❖ Processor type: Make certain you are using a processor with adequate speed and that it isn't being overused.

System Configurations

Caching and accounts are two places where configurations are key to performance tuning.

- ❖ Any application that needs an external connection for use (for example, across the network) will need a cache. The type of cache will depend on the application in use.

Search caches, reverse proxy web caching and page caching. In addition, the `cacheInclude` Idoc function can be used to cache resource includes. See the *Managing System Settings and Processes Guide*, *Idoc Script Reference Guide* and the *Reverse Proxy Resource Guide* for details about these cache options.

A poorly configured cache setting can cause your web server to continually go back to the origin server for content, often needlessly if the content hasn't changed. If the cache is too small, under-used or ignored, your content retrieval time can be slowed considerably. One of the first areas to consider when optimizing performance is the cache configuration for your system.

- ❖ If your Content Server has been configured with too many security groups, system performance can be slowed. In general, if you need more than 15 security groups, you should switch to an account-based security model. If accounts are enabled, you should only need a maximum of 15 to 20 security groups. Accounts can be set up in a hierarchical structure, enabling some users access to an entire branch of the structure while limiting permissions for other users. This can greatly speed system response time, especially during searching. See the *Planning and Implementation Guide* and the *Managing Security and User Access Guide* for details on setting up accounts and user permissions.

OTHER QUESTIONS TO ASK

Answers to the following questions can help you narrow down areas of focus for your performance tuning.

- ❖ Have you outgrown your Content Server? As content items are added to the system, queries to the database and the search collection tend to take longer. One way to handle this is to spread the content items across multiple content servers. You may need to add another server in a cluster configuration to alleviate slowdowns.
- ❖ Has your site traffic increased? Site usage is defined as the number of requests to the Content Server but the load from different types of requests (static vs. dynamic) is highly variable. Site usage also includes the number of users on the site combined with the frequency and duration of site visits. If the traffic is high, you may need to add another server. That may alleviate slowdowns.
- ❖ Is your network connection adequate? Like caching, your network infrastructure can affect the speed at which content is delivered.

- ❖ Are you using the right version of the Content Server? Newer versions may have optimized enhancements built in to the core software which cannot be duplicated with older versions. Consider upgrading your software if needed.
- ❖ Could add-on components be affecting performance? If you have a large number of add-ons with an older version of Content Server, you may benefit from upgrading your version. Many add-ons that have been added to the core Content Server software have been optimized for use.

Additional questions to ask are:

- ❖ Is the memory allocated to the Content Server adequate?
- ❖ Does the database have enough memory?
- ❖ Are the indexes on the database tables optimized?
- ❖ Is the database properly tuned?
- ❖ Does the search database have indexes on the fields that are being searched?
- ❖ Is the search database optimally organized?
- ❖ Are there any inefficient search or database queries?

STEPS IN THE TUNING PROCESS

Optimization is an iterative process with six basic steps:

1. Identify and evaluate the problem. Check system usage and determine what is acceptable and unacceptable behavior for your site. See [Analyzing the Results](#) (page 3-7) for some general system statistics that you can use to evaluate your own site's performance.
2. Design your benchmarks. A benchmark is a baseline against which you'll judge your testing results. Benchmarks help you quantify performance and assist in focusing on the correct area.
3. Run your benchmarks. Develop a script that mimics a typical user's session (for example, navigating to certain pages, submitting forms, checking in content) and use that as your benchmarking scenario.
4. Analyze the results. Review the test results and see if you can determine where the slowdown is occurring.

5. Apply optimization techniques. Different techniques can be used depending on the area where the bottleneck occurs. You can also optimize those areas outside of the content server, such as the database, the indexer, and the system hardware.
6. Re-test. Run the tests again to determine if the optimization techniques have been useful. Record the new performance data and watch for any side effects. Sometimes a performance improvement in one area can cause a slowdown in another.

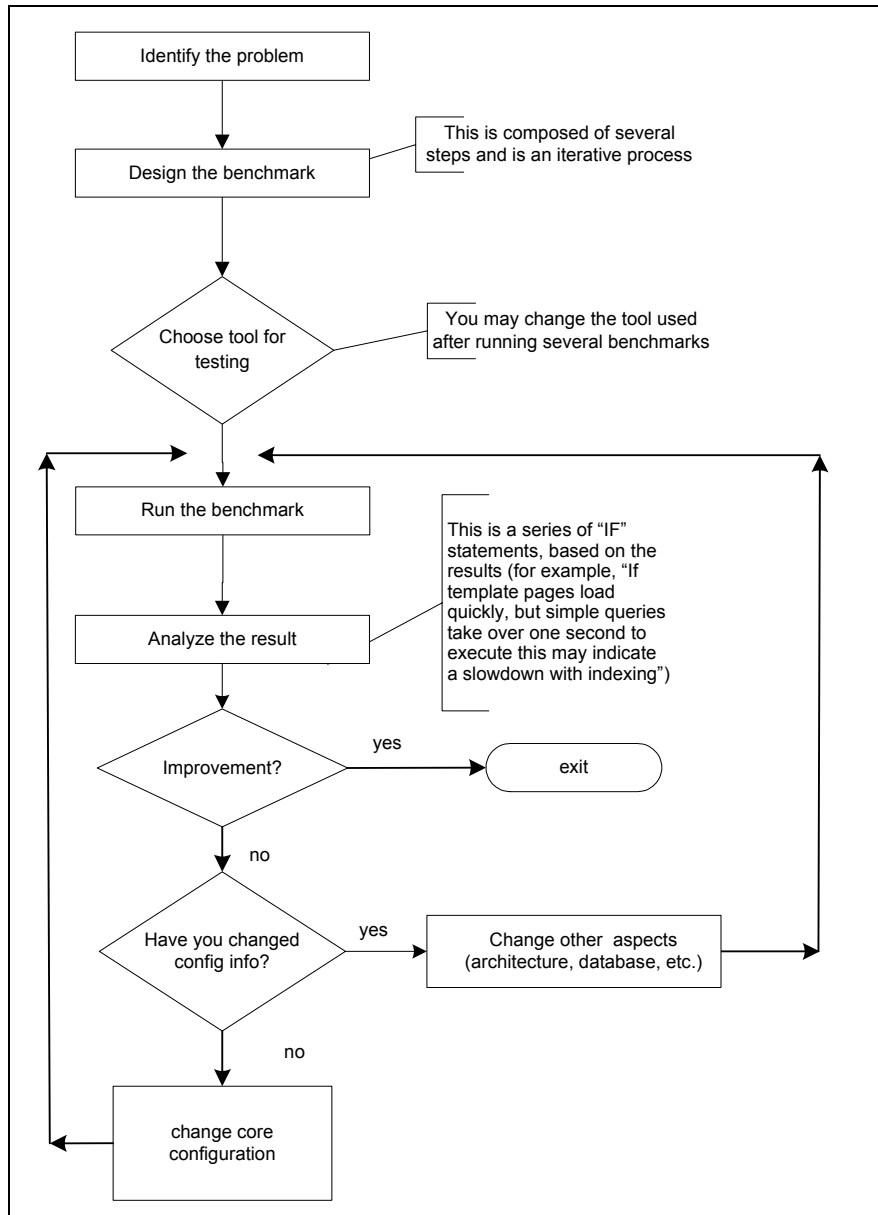


Note: Keep in mind that the improvements seen on the test server may be misleading unless the benchmark was a 100% copy of a real-world scenario. Do not expect the same level of performance improvement on a real-world system.

The remainder of this document discusses these steps in more detail.

The following flowchart describes, in general terms, the steps used in the optimization process.

Figure 2-1 Optimization process



BENCHMARKING

OVERVIEW

Benchmarking is the process of designing tests to document system behavior and refining those tests to help you narrow down where problems may be occurring. This chapter discusses the following topics on benchmarking:

- ❖ [Designing Your Benchmark](#) (page 3-1)
- ❖ [Benchmark Tools](#) (page 3-4)
- ❖ [Running the Benchmark](#) (page 3-6)
- ❖ [Analyzing the Results](#) (page 3-7)
- ❖ [Analysis Tips](#) (page 3-11)

DESIGNING YOUR BENCHMARK

A benchmark is a test used to compare performance of hardware and/or software. When comparing benchmark results, it is important to know exactly what the benchmarks are designed to test. A benchmark that tests graphics speed, for example, may be irrelevant to you if the type of graphical applications you use are different from those used in the test.

Basic benchmarks can be designed quickly, but more detailed benchmarks take time and thought to develop. It's important to remember that your benchmarks should be designed to quantify the performance for the average user. There are always exceptions to the average, but focusing on the average user will have the most benefits.

Before beginning the design process, keep the following points in mind:

- ❖ Make the benchmark as ‘real world’ as possible. Try to use your production server or use an exact replica of the production server. This allows you to mimic what users are seeing.
- ❖ Isolate the benchmark server. Only allow testers to have access to that server. Note that this isolation is required to quantify performance changes but it may skew results because isolation does not mimic the real world.
- ❖ Document your process. Verify all software and hardware that is running, and make sure that the documentation of your performance is continually updated.

Benchmark Design Points

The following considerations should be used when designing your benchmark.

Review and Document Your Current System

Make sure that you document all details about your system before beginning the benchmark process. This includes the following points:

- ❖ Check that there are no errors in any of the customizations done for your system. Logging errors can slow down a system even though they may not cause a stoppage. Fix any IdocScript errors in customizations before running your benchmarks.
- ❖ Make sure all current patches are applied for all parts of your system, including the operating system, hardware, database and security system as well as the Content Server.
- ❖ Document the setup information in place at the start of the benchmark:
 - Hardware: The size and configuration of the CPU, the memory, manufacturer and any other pertinent details.
 - Software: What software and version is running for your database, web server, and your Content Server. Note any customizations that have been done?
 - Network: Document any network hardware that is involved in the benchmark process. This could include hubs, switches, firewalls, proxy servers and other network devices. Be sure to note any bandwidth limitations of the network hardware.
 - Security: Document all aspects of your security, including LDAP and Active Directory as well as any Content Server security groups and accounts. Continual security checks can slow performance.

- ❖ Document the usage information for your system. Note the following information:
 - How often is batchloading run?
 - Is publishing run on a daily, weekly or as-needed basis?
 - Are most of your users contributors or consumers?
 - How often is archiving done?

Documentation can sometimes point out places where problems arise before the benchmark is even run. After examining your system documentation, determine what exactly needs to be fixed.

- ❖ Are you trying to increase the speed of delivery of web pages?
- ❖ Are you trying to maximize the number of users who can contribute to the site?
- ❖ Do you want to increase the number of web pages that can be delivered?
- ❖ Do you need to minimize the search and check-in time?

Mimic a Typical Session

As mentioned before, your benchmarks should be set up to test the results for a normal user's session. Design a web script for a typical user's session:

- ❖ Navigate to common pages (Site Studio, publisher, or standard templates)
- ❖ Perform searches
- ❖ Perform an update on a piece of content
- ❖ Start or participate in a workflow
- ❖ Check in content

Choose 20 - 30 pages of content of varying complexity to use for this script. Then plan to repeat the test 1000 times, to determine the timing.

If you are testing batchloader or the archiver, get a sample of typical items for a batch test. This will help you test the database, indexer, and refinery. If you are testing the Publishing Utility, get typical examples from Site Studio, Publisher, and the Connection Server. If possible, get 1000 typical items from each in order to get a good test result.

BENCHMARK TOOLS

After you have designed your benchmark script and determined the area of focus, choose a tool to use for the benchmarking.

Many different tools are available. Simpler tools include the `taskmgr` command on Windows or the `top` command on UNIX system. Other tools include Mercury LoadRunner, the Microsoft Application Center Test (.NET) and various Java profilers.

Many database analysis tools are also available, most specific to a particular database. Check with your database analysts on site to determine which might be best to use.

In addition to tools that monitor your system, you can use several Content Server tools to monitor timing and performance. On the Admin Applets page, check the Administration Log files for Refinery, Archiver, and the Content Server.

You can also enable tracing by selecting different trace actions on the System Audit Info page, available from the Administration tray. Several different tracing reports can be generated, including ones for Indexer, socket requests, system database, and others. See the *Troubleshooting Guide* for details about enabling the tracing reports available in the **Edit Active Console Output Tracing** section of the System Audit Info page.

The following is a list of all available tracing options, some of which are unrelated to performance tuning:

- ❖ **applet**—This trace contains result sets from initialized applets, such as the Configuration Manager or User Admin.
- ❖ **archiver**—This trace provides information about archiving activities, including the reading and writing of archiver data files and the time the activities were initiated and finished.
- ❖ **archiverlocks**—This trace provides information about the locks put on files during archiving activities, including time initiated.
- ❖ **chunkedrequest**—This trace displays the messages and headers that are created when large requests are ‘chunked’ into smaller requests.
- ❖ **docprofile**—This trace displays the computation of content profiles, specifically the evaluation of the rules that determine which fields are labels, hidden, and so on.
- ❖ **encoding**—This trace provides information about encoding transformations that have occurred and the activities where encoding occurred.

- ❖ **filelock**—This trace displays information about short-term system locks put on directories (during activities like archiving, for example) with a focus on collisions that occur and timeouts.
- ❖ **filelonglock**—This trace displays information about the creation, removal, and maintenance of long term locks imposed by the system.
- ❖ **filequeue**—This trace displays information about accesses to a file queue.
- ❖ **indexer**—This trace displays information about index functions that occur when the database is updated, including the steps taken to update the index and the time elapsed for each step.
- ❖ **indexermonitor**—This trace provides a brief summary of automatic index activities, including time started and ended.
- ❖ **indexerprocess**—This trace displays information about a manually launched index process and indicates if the process terminated properly.
- ❖ **localization**—This trace displays information about localization usage and activities.
- ❖ **mail**—This trace provides a description of mail sent by the content server.
- ❖ **pagecreation**—This trace displays information about the creation of displayed pages, including the server thread and the time taken to generate the page.
- ❖ **requestaudit**—This trace provides summary reports on service requests, including the elapsed time for the requests and the number of requests made.
- ❖ **scheduledevents**—This trace provides a list of hourly or daily background scheduled events.
- ❖ **schema**—This trace provides information about schema publishing (tables and views published as .js files) and caching (tables cached into content server memory).
- ❖ **searchquery**—This trace displays information about recent searches, including the fields used to search on and the order of sorting for results.
- ❖ **socketrequests**—This trace displays the date, time, and thread number of socket requests as well as the actions during the request.
- ❖ **system**—This trace displays internal system messages, such as system socket requests and responses.
- ❖ **systemdatabase**—This trace provides information about database activities, including queries executed, index updates, threads used, and time initiated.
- ❖ **transfermonitor**—This trace displays information about the archiver and the batch file transfer activities.

- ❖ **userstorage**—This trace describes the access of external user repositories, including what actions were taken during access.
- ❖ **workflow**—This trace displays a list of metadata on content items going through workflow, including document title and revision number.



Note: The tracing reports are available on the System Audit Info page in Content Server 7.0 and later. For earlier versions, you must enable tracing by using configuration flags. See the *Idoc Script Reference Guide* for configuration details.

RUNNING THE BENCHMARK

As you run the scripts you developed you should monitor the processes on all servers. Verify the CPU and RAM usage as well as the I/O reads. Check Content Server processes, also, such as `htmlexport` and Java processes. Also check any processes that are used by the database.

Also check the resource usage while you run the tests. This type of examination is dependent on the operating system in use, but when doing performance testing it is a good idea to shut down processes that may be CPU-intensive or if they might access Content Server resources. Included in this is to check if virus scanners are running or if other processes are trying to access the server. Verify if another process is using resources that the Content Server depends on — file server, database, or web server. This could affect performance results.

After running the initial benchmarks, adjust your scenarios to get a wider range of performance figures:

- ❖ In the best-case scenario, turn off all logging and do not allow contribution. Run a search or a check-in and clock the time used. This will give you the fastest time possible.
- ❖ In a bad-case scenario, turn off the search cache and re-run the test scripts.
- ❖ In a worst-case scenario, run the batch and web tests simultaneously. This will give you the worst possible results.

While these types of scenarios don't often happen in real life, it's important to know what is the best that can be expected and what is the worst. For example, even though it may be scheduled for an off-peak time, an emergency batch operation could slow your system considerably. It's important to know what is possible.

As the benchmark tests run, gather as much timing data as you can. Analyze the Content Server logs for timestamps, and determine the number of requests per second per user for the web site tests. Once you know the timing numbers, you can begin to analyze the results.

ANALYZING THE RESULTS

The first step in result analysis is to determine if your results are slow. Typical goals for business content manipulation (average content of average size) are:

- ❖ Read-only requests: 20 requests per second per GHz per CPU. Note that this is not a hard-and-fast rule. Some read-only pages can take up to ten seconds to render, usually because the page is assembling pieces of content from a diverse set of resources. A read-only request figure is dependent on the work necessary to create the page.
- ❖ Raw queries: Four per second per GHz per CPU for simple searches. This assumes that you are not using a search cache. Performance is influenced by your security model, CPU, file system, query complexity, and the data organization. Therefore, this figure can vary widely.
- ❖ Contribution: Four check-ins per second per GHz per CPU. Again, this is assuming average business content and not complex content that may require more time. Contribution performance is primarily limited by the database and by network performance, in addition to file size.

If your results are slower than those listed, you should consider performance tuning. You may need to enable more trace reports on the System Audit Info Page in order to narrow down where problems lie.

The initial benchmark results provide pointers to possible problem areas, as described in the remaining sections in this chapter.

Search and Index Problem Areas

Search and index are related even though these are separate functions. In many cases, indexing optimization is undertaken not to speed up indexing but to create indexes that result in faster searches.

Indexing-specific performance problems are usually related to large documents or encrypted or unsupported formats. A way to verify this is to check for errors in log files then use debugging to try to clear up the problem.

The following benchmark results may point to a possible problem with the Search or Indexer function:

- ❖ If template pages load quickly, but simple queries take over one second to execute.
- ❖ If navigation pages are executing highly complex queries. Analyze the `searchquery` trace to determine what is being executed. Note that dynamic navigation can use either the search collection or the database, so this may not be a clear problem.
- ❖ If the system has a complex security model. A complex security model can add to search time by forcing the system to evaluate a user's request and security check unnecessarily.
- ❖ If there is a large amount of general contribution, such as user checkins, external portal application checkins, automated checkins, archive imports, and publisher checkins. Search caching and other performance issues are affected by the amount of contribution coming into the system.
- ❖ Frequent uses of the Batchloader can also cause slowdowns because the search cache has trouble maintaining utility during large swaps of content. Batchloading is unique in the amount of contribution it creates and the length of time it takes, although the times when it is used can be controlled. In many systems that use Batchloader, most of the content in the Content Server comes from the Batchloader.
- ❖ If the search cache is under-utilized. Check the cache statistics on the System Audit Information Page. If there is a small cache size or few hits, the search cache may be under-utilized. If the hit ratio is below 50% it will be worthwhile to consider areas to optimize. It is difficult to achieve a hit ratio above 75% on a consistent basis unless a simple security model is in use and there are a limited number of queries.
- ❖ If the search collection is not local. A slow network file system will slow search results.
- ❖ If there is massive contribution and consumption on the same server.

See [Index Tuning](#) (page 4-2) for details about index optimizations.

Database Problem Areas

The following benchmark results may point to a possible problem with database usage:

- ❖ If there are many connections in a long queue, the database query may be taking too long. Check the System Audit Information Page for a list of active and waiting connections. If there are threads waiting for a connection or queries taking a long time

to process then a problem exists. Commonly issued queries taking longer than two seconds are typically a source of performance issues.

Some queries do take a long time. For example, some indexing queries, workflow queries or reports may take a relatively long time to execute. These are usually processed on the background threads, however, and are rarely executed. Even so, they should not take more than a minute to execute.

- ❖ If there are dropped connection errors or too many connection errors, there may be a problem. Check web logs for connection errors.
- ❖ If most of the request time is spent in the database or if the database CPU is overworked, this can slow database response. Monitor the `systemdatabase` tracing report to check the timestamps on queries.

See [Database Tuning](#) (page 4-5) for more details about database tuning.

Network Problem Areas

The following benchmark results may indicate a bottleneck in the network infrastructure:

- ❖ If the remote connections, specifically those to the database, are slow but the remote services are under-used. You can use the `trafshow` utility or `Ethereal` to determine the network saturation.
- ❖ Compare running the query directly on the database and across the network. If there is a significant disparity in time, the network connection may be to blame. You can also use the `systemdatabase` trace to check for timestamps on queries.

See [Chapter 6 \(System Architecture Tuning\)](#) for details about network tuning.

Core Engine/JVM Problem Areas

If publishing or page rendering is slow there may be a problem with the core Content Server configuration. Check the default configuration for your Content Server and see if there are areas to be changed. See [Adjusting Configuration Flags](#) (page 4-1) for details.

If the following conditions exist, there may be a Java JVM problem:

- ❖ If the system is rendering a large amount of HCSP or JSP files.
- ❖ If the system is a consumption server.
- ❖ If a large amount of workflows or subscriptions are run.

- ❖ If the Java VM uses all of the CPU or memory even if searching isn't performed. It's possible that multiple CPUs are not being fully utilized. You may need to tune the Java VM to recognize the existence of other CPUs.

See [Java Tuning](#) (page 4-8) for more information about JVM optimization.



Note: JVM does not produce reliable results with small benchmark tests. A larger benchmark run may help reveal if the bottleneck is really caused by the JVM.

File System/Memory Problem Areas

The following benchmark results may point to a file system problem:

- ❖ If there is larger than expected file activity. Monitor the I/O reads to determine how much swapping is being done. A large amount of swapping indicates an insufficient amount of RAM. A method to determine the amount of I/O swapping is dependent on the operating system in use, but in general if the normal RAM usage is greater than the physical RAM then too much swapping is occurring.
- ❖ If the file system is nearing capacity, slowdowns can occur.
- ❖ If the file system is remote but there is not a high performance network in place. In general, SAN is very fast, expensive and can require long setup times while in comparison some NAS are slower, inexpensive, and easier to install. SANs typically do not allow two systems to access the same file while a NAS does. Because of that, a NAS or other network file system is required for applications such as clustering. A NAS is often used so multiple Windows machines can use the same SAN.

Some file system problems can suggest an underlying network problem. The two are often linked. See [Chapter 6 \(System Architecture Tuning\)](#) for information about file system tuning.

Web Security Filter Problem Areas

The following are indications that there may be a bottleneck in the web security filter:

- ❖ If the user's initial login takes several seconds to accomplish but subsequent page delivery is fast.
- ❖ If many users log in to the browser's home page but do not navigate further. This causes a high filter load initially, causing slowdowns. Monitor the verbose version of the `userstorage` trace report for timestamps.

See the *Managing Security and User Access Guide* for information about internal and external security and the configuration options which can be used for tuning.

ANALYSIS TIPS

Keep the following points in mind when analyzing your benchmark results:

- ❖ Java profiler reports can be misleading. Do not try to analyze Java profiler reports without an extensive understanding of the Content Server and a good understanding of JVM.
- ❖ Logging and virus scanners can slow a system. Turn off all loggers and virus software before running your benchmarks.
- ❖ Try to establish real-world benchmarks and determine if your system is consumption-driven or contribution-driven. Turn off the search cache and watch the timing of various operations. This will give you a good idea of where to focus your tuning efforts.

TUNING THE CORE CONTENT SERVER

OVERVIEW

After running your benchmarks and analyzing system slowdowns, you can often attain good performance benefits by some simple tuning of core Content Server behavior. This chapter covers the following topics on tuning the core Content Server:

- ❖ [Adjusting Configuration Flags](#) (page 4-1)
- ❖ [Java Tuning](#) (page 4-8)
- ❖ [Other Tuning Options](#) (page 4-9)
- ❖ [Idoc Script Caching](#) (page 4-5)
- ❖ [Index Tuning](#) (page 4-2)
- ❖ [Database Tuning](#) (page 4-5)

ADJUSTING CONFIGURATION FLAGS

The simplest way to adjust performance tuning is to adjust the default configurations which are shipped with the Content Server. The following list summarizes several variable that can be changed. For complete details and additional configuration variables, see the *Idoc Script Reference Guide*.

- ❖ **Indexer variables:** Several variables can be used to change the way the indexer operates as well as what kinds of renditions are indexed. See Chapter 5 of the *Idoc Script Reference Guide* for a complete list of variables that affect indexing.

It's important to maintain a clean, small index wherever possible to optimize consumption. However, this may not always be optimal for contribution. Always consider a balanced approach based on the usage at your site.

- ❖ **Search variables:** It's important that the number of search connections be kept to a minimum. Generally it's best to have one connection per CPU. The goal is maximum throughput of the system. For any given hardware, software, and network configuration there is a value for search and database connections that gives the best throughput. The following variables can be used to set the number of connections and the cache behavior.
 - `CachedResultRowCount`: Sets the size of the search cache. The search cache can improve search performance by caching recent search engine queries. Increasing the size reduces search time, but increases memory usage.
 - `MaxSearchConnections`: Sets the maximum number of search connections that are open at one time
- ❖ **Database variables:** Where possible, keep the number of database connections to a minimum. Use the `NumConnections` variable to set the number of open connections to the database. The default is 5, which can be low if the database will be exercised heavily under load. This setting applies to the content server and any stand-alone applications and utilities, so each application will use the specified number of connections.
- ❖ **Other variables:** Use the following variables to set queue and credential information.
 - `IdcServerSocketQueueDepth`: Specifies the depth of the TCP/IP socket queue.
 - `UserCacheTimeout`: Sets the timeout in milliseconds for the temporary cache of global and external user information.

INDEX TUNING

The Content Server uses a variety of indexing tools such as Verity, FAST, or different database index tools. Each indexing tool provides full-text indexing (in which every word in a file is indexed) and metadata-only indexing. Full-text indexing takes longer than metadata indexing but it can return a more comprehensive result. The indexing tool to be used is chosen prior to installation based on the purpose for the content server and the environment in which it will perform.

You can optimize the index and search functions. This section provides an overview of search index use. See the *Managing System Settings and Processes Guide* for more details.



Note: Index tuning is heavily dependent on several different factors. There is no ‘one size fits all’ recommendation for performance tuning of indexes. It is recommended that you contact Support before attempting any extensive performing tuning with your indexes.

Working with the Search Index

Administrators can use the Indexer tab on the Repository Manager screen to update, rebuild, configure, or disable the search index and search collection.

- ❖ To update the search index, select the Indexer tab on the Repository Manager. In the Automatic Update Cycle area, click **Start**.
- ❖ To rebuild the collection, select the Indexer tab on the Repository Manager. In the Collection Rebuild Cycle area, click **Start**.
- ❖ To configure an update or rebuild, select the Indexer tab on the Repository Manager. Select **Configure** in either the Automatic Update Cycle portion of the screen or in the Collection Rebuild Cycle portion of the screen. Specify the number of content items per indexer batch and the number of items per checkpoint. The number of items per batch is the maximum number of files the indexer will process at one time; the checkpoint number is the number of files that will go through all relevant indexing states at a time. Also specify the debug level. See the *Managing Repository Content Guide* for details.
- ❖ To disable full-text indexing on specific files, define a File Format named **application/noindex** in the Configuration Manager. On the System Properties Options tab, select **Allow override format on check in**. Instruct users that when they check in a file that they do not want indexed, they should select **application/noindex** as the file format. However, changing the file format keeps the browser from being able to open the file automatically.

The `SelectivelyRefineAndIndex` component can be used to control which documents are converted and which are indexed. Contact Consulting Services for details on its use.

- ❖ The `IndexerLargeFileSize` variable can be used to set the file size in megabytes at which the Indexer will place the file into its own batch. Files larger than this setting will be indexed in a separate bulkload.
- ❖ To allow new documents to be added to the collection during a rebuild cycle, set the `AllowConcurrentUpdate` configuration variable to `True`.

About Full-Text Indexing

When full-text indexing is applied, every word in a file is indexed, not only its metadata. See the *Managing System Settings and Processes Guide* for details about supported formats and more details about the full-text indexing tools described here.



Note: A full-text index is often called a “Search Collection” or “Search Collection Index”. A content server instance uses a single index collection. When you add documents, they are added to the same collection. When the collection is rebuilt, the content server creates a new collection. Only one collection is active at any time.

Indexing with Add-On Integrations

Verity Integration and FAST are add-on indexing integrations which work with Content Server, regardless of the database used. With these tools you can choose to apply full-text indexing to all converted files with a specified set of formats. You can enable contributors to specify if a file is indexed with full-text indexing by enabling the format override option in System Properties, as described earlier.

When users search for content by metadata or keywords, a query is issued against the search index, not the database. The results can be sorted by any of the metadata fields or based on a relevancy score assigned by the search engine.

See *Choosing a Search Solution* for more details about using these indexing add-ons.

Database Full-Text Indexing

In order to use database full-text indexing, add the following line to the content server’s `<install_dir>/config/config.cfg` file:

```
SearchindexerEngineName=DatabaseFullText
```

To use metadata-only searching use:

```
SearchindexerEngineName=Database
```

Do not use both settings in the same system.

As with other tools, full-text indexing is applied by default to all files converted to specific formats. See the *Managing System Settings and Processes Guide* for a list of accepted formats.

IDOC SCRIPT CACHING

Idoc Script is evaluated on the server side, not the client side. Therefore, page elements are processed after the browser makes a request but before the requested page is returned to the client.

One of the most time-consuming activities performed within Idoc Script pages is performing service calls within the context of the page. This involves extra processing for the server when performing activities such as search and database queries. The Content Server will perform caching on some queries, but there may be situations in which the cache may not be as valid as desired due to content activity.

Idoc Script caching can be used to encapsulate individual includes and store the cache on a per-user basis or for the application as a whole for a specified amount of time.

The `cacheInclude` configuration variable can be used to include a `dynamichtml` fragment from a cache. This is useful for fragments that are used multiple times, such as a global cache, named cache, or session cache.

The `setMaxAge` variable can be used for caching an entire page. It uses `cache-control` HTTP headers for page caching. This is useful for pages which are used frequently.

DATABASE TUNING



Important: Database tuning is a separate issue from database full-text index tuning. Database tuning involves optimizing your database software. Full-text index tuning involves working with the search collection and optimizing it for use with your database.

Several methods can be used to tune your database to work with the Content Server.

- ❖ Optimize file I/O performance, particularly for indexes. A solid-state disk can be useful for this.
- ❖ Add extra indexes to commonly searched columns. To determine which columns to index, use the `systemdatabase` tracing report from the System Audit Information Page. This will show which columns are typically searched.
- ❖ Make sure that custom metadata fields are indexed if they are used for searching. Use full-text indexing for searches using the `contains` keyword, or normal indexing for search that simply match.
- ❖ Update your database statistics as your database grows, especially before and after inserting large amounts of content. This helps ‘train’ your database on what is

commonly used. The process for this depends on the particular database. SQL Server does this by default, for example. Consult your database documentation for details.

- ❖ Optimize the row-lock algorithms in your database. For example, Oracle makes assumptions about database use that may not work correctly with Content Server. When using Sybase, you can use the SQL script provided in your `<install_dir>/database/Sybase/admin` directory to optimize row-locking.
- ❖ Consider creating custom services or components that are specific to your site's needs. For example, you may want to create database-specific queries for better performance or create services that simplify the checkin of large batches of content. See the *Working with Components* guide for details.
- ❖ The Oracle Query Optimizer component can be downloaded from the support site. This component is designed to improve your Oracle database's performance by removing inefficiencies in user queries. With this component you can add hints to queries that force Oracle to perform searches more efficiently. In addition, this component maximizes indexing capability in Oracle.
- ❖ When using Oracle, you should also consider the following tactics:
 - Try to improve the responsiveness of queries. By using the following command, you can optimize the query to get the first row back as quickly as possible before completing the query:

```
optimizer_mode=first_rows
```
 - Consider using table partitioning. The size of the partition depends on your data distribution and hardware configuration. If using a version of the Content Server prior to 7.5, do not partition the DocMeta table unless your metadata structure is very stable.
 - Set `Cursor_Sharing=SIMILAR`. The default is `Cursor_Sharing=EXACT`. This change is always recommended.



Note: The information discussed here is specific to the Content Server. For other database tuning options, consult your database documentation.

General Database Maintenance

Regardless of the type of database used, there are several maintenance plans you can implement to maintain optimal performance. This section provides a summary of the types of maintenance that should be done periodically. For complete details, consult your database documentation.

Backup Planning

An adequate backup and recovery plan can be critical to maintaining the health of your database. The type of backups needed for different databases will vary. Databases that are static in nature may not require frequent backups but instead only a nightly or weekly full backup. A backup may not be required for databases that can be easily reconstructed from source transactional databases. Backups should ideally be performed as quickly as possible to minimize performance effects. However, some backups require exclusive access to the database while the backup is running. This means that no users can be connected to the database and no work can be performed.

In general, backups should be performed when the impact on database work is minimized.

System Maintenance

Database performance relies on several key maintenance operations which should be monitored and, if necessary, adjusted:

- ❖ *Space*: Different databases have different system-level space structures, such as tablespaces and database space, which should be monitored. Performance can be improved when unusual database space or tablespace is reclaimed and reorganized.
- ❖ *Database integrity*: Database integrity ensures that the underlying file structures in the database are available for use, that the databases are architecturally intact and that the data dictionary is consistent with itself and the accompanying database objects. Different databases use different commands and tools to perform this validation.
- ❖ *Memory*: Just as a table or index can become fragmented through use, memory used by a database server can also become fragmented. Many memory maintenance functions have drawbacks, however, because the elements of various caches must be reloaded from disk. Therefore, performance may initially suffer until frequently used objects and data are restored to memory.

Other Database Maintenance

Databases are comprised of tables and indexes, as well as other objects. There are a wide range of object-styled maintenance operations that should be run as part of planned maintenance:

- ❖ *Space fragmentation*: Almost all database engines offer reorganization capabilities for tables and indexes which have grown and shrunk over time.
- ❖ *Table reorganization*: Tables sometimes need to be reorganized when a row in a table needs to expand into a block of space but can't due to a lack of free space. The row is

relocated to another block or page with enough free space, leaving behind a record pointer. When the row is needed by the database, the pointer must be read then the row itself is read. Tables such as these can adversely affect performance because the database engine must do extra I/O work in order to read the appropriate row.

- ❖ *Index reorganization*: Indexes with heavy activity can become fragmented, resulting in longer scan and navigation times. Index rebuilding should be done on a regular basis to help achieve good performance results.

In addition, two other aspects should be considered when performing database maintenance. A *plan dependency list* takes into account the fact that parts of a database maintenance plan depend on the successful execution of other parts of the plan. For example, backups would be run only if an integrity check shows that the database is correct. The scheduling of backups is dependent on a valid integrity check.

Another aspect is the use of *notifications*. If a plan fails to execute when scheduled or if unexpected results occur, a notification will be sent to the appropriate personnel so the operation can be checked.

JAVA TUNING

The Java Virtual Machine (JVM) is self-tuning, but you can add some tuning parameters to the Content Server's `intradoc.cfg` file. For example, you can use the following setting in the `intradoc.cfg` file to optimize the memory and the Java compiler:

```
HEAP_OPTIONS=-Xmx512m -Xms512m -server
```



Note: Only use the `-server` option when you do not plan to use standalone applets.

You can increase the 512m value to 1000m or 2000m if there is enough RAM available. This is usually needed only if the size of the search cache has been increased or if Folders caching is used and there are many folders.

A Java “out of memory” error is fatal. If that occurs, you will probably need to increase the `Xmx` value.

When using multiple CPU systems, you should use Java version 1.4.1 or later and tune your garbage collector for multiple CPUs. For more information about tuning Java applications and for Java documentation, see <http://java.sun.com/docs/performance/>.

OTHER TUNING OPTIONS

Several other changes can be made to the core Content Server functionality to help tune the performance.

- ❖ You can simplify the pages which are delivered from Publisher or Site Studio. Some pages do multiple requests per page, and many of the requests are not needed. Reduce the number and complexity of queries and avoid queries that do not cache, such as timestamps or navigation queries.
- ❖ In Content Server 7.5 and later, you can use database search to search for exact matches. This should only be done after tuning the search performance and if the database is not also loaded.

To do so, set the following parameters to the GET_SEARCH_RESULTS service:

```
SearchEngineName=DATABASE
```

To use the database search function, set the QueryText parameter to specify what text to search for:

```
QueryText=dDocType LIKE 'ADACCT'
```

This allows you to use SQL-like queries to search the database directly.

- ❖ Avoid using the eval function. Use setResourceInclude or setValue instead. setResourceInclude allows dynamically constructed script to be assigned to an include. setValue is used to set a value to a target area based on a key.
- ❖ Eliminate full text searching or consider using conditional full-text indexing (a customization). Full-text searching uses more resources and can be time-consuming. If full-text searching is not required, metadata only searching could be implemented. This reduces overhead because cached searched can be leveraged.

Alternatively, a different Content Server instance can be used to separate content requiring full-text indexing from that not requiring such indexing. This relieves some of the performance load but this option does require more licenses.

- ❖ The Dynamic Converter tool can be used to convert content to HTML as needed, or during checkin. Use of Dynamic Converter in a portal integration deployment can remove the high volume of active content changes in a consumption environment (when using Site Builder/Publisher to translate and republish content into the active consumption environment).
- ❖ Try to reduce the number of checkins that occur during high-peak hours and confine publishing with Content Publisher during hours of off-peak usage.

- ❖ Another way to reduce search transactions is to pre-generate navigation to content. Generating navigation to content for queries as needed produces more search transactions in Content Server. Some navigation entries may be pre-generated, which reduces the number of search transactions happening to the system.
- ❖ You can also use portlets to publish content from the Content Server to a file system using Content Publisher and have the portal retrieve content from the published location. This can improve performance by eliminating dependencies between the portal application and the Content Server. The full-text searching will have to be managed by the portal. In addition, the portlets will have to be rewritten to local and display content from a static site.
- ❖ Examine the security configuration at your site. The more complex the security model, the longer the query that is created and the longer the query will take to run. For example, if a user runs the following query:

```
dDocName <matches> 'test'
```

The actual query that is run is:

```
(dDocName <matches> 'test') <and>  
(dSecurityGroup <not> 'Secure')
```

If you have ten security groups and every user has access to five of those ten groups, then your searches will be needlessly complex as all security groups are examined during a search.

- ❖ Check the Web Server Filter in use at your site. The Content Server requires a web server to deliver pages through a browser. A web server filter is installed in order to authenticate user requests. You can change Web Filter options by clicking the **Filter Administration** option from the Administration tray.

The options on the Configure Web Server Filter page can be used to disable gzip compression, set the type of authentication used, set the cache timeout value, and enable logging of data and headers sent between the filter and the Content Server.

See the *Managing Security and User Access Guide* for more details about configuring the Web Server Filter.

TUNING ADD-ON COMPONENTS

OVERVIEW

After tuning the core parts of the Content Server, you may also want to turn your attention to the add-on components to the Content Server. This section covers the following topics about tuning add-on components:

- ❖ [Batchloader](#) (page 5-1)
- ❖ [Dynamic Converter](#) (page 5-2)
- ❖ [Site Studio](#) (page 5-2)
- ❖ [Publishing Utility/Connection Server](#) (page 5-3)
- ❖ [Inbound Refinery](#) (page 5-4)
- ❖ [Content Publisher](#) (page 5-4)
- ❖ [Content Integration/Portal Suite](#) (page 5-5)
- ❖ [Tools to Assist Optimization](#) (page 5-5)

BATCHLOADER

The Batch Loader utility is used to insert, delete, and/or update large quantities of content on your content server at one time. The following suggestions can help improve the performance of your system when you are using the batchloader:

- ❖ On smaller systems, temporarily disable other activities before initiating a batchload. For example, disable the auto-update indexer cycle, the Inbound Refinery, and the

PDF Converter. These functions will all run automatically after the batchload, so disabling them before initiating a batchload will save time.



Note: Disabling other activities is not necessary on large, multi-processor systems which can take advantage of concurrent processing. In those cases, do not turn off background activities.

- ❖ If using Verity, optimize the search collection prior to a batch insertion. See [Index Tuning](#) (page 4-2) for more details.
- ❖ Analyze your database usage during a batchload operation using tools provided with your database. See where you might be able to improve the database query optimizer. You may want to do a small batch insertion first to verify the database usage.

DYNAMIC CONVERTER

Dynamic Converter is used for on-demand publishing of native business documents to HTML, XML, or wireless formats.

A simple way to speed Dynamic Converter functioning is to use templates. To do so, select **Template Selection Rules** from the Dynamic Converter Admin menu. A template rule determines the conversion and layout. On submission, the content is converted and cached. Subsequent users retrieving the content are retrieving an already-converted file and do not have to wait for it to go through conversion.

SITE STUDIO

Site Studio is used to automate Web site creation and enable content contribution in a WYSIWYG form-based environment. Several different methods can be used to enhance Site Studio's performance.

- ❖ When using native content to generate your web pages, Dynamic Converter is used in the background to convert the content. Therefore, using Template Selection Rules to pre-convert your content will result in faster response time.
- ❖ Optimize queries in dynamic lists when they are used to deliver content. Try to keep queries simple. Avoid matching on time stamps in addition to date stamps. Time stamps are almost impossible to match.

- ❖ Optimize services that are executed on pages. Check how many services are running and make sure that only the ones running are those that are necessary to page delivery. Remove any unnecessary services, work, or data.
- ❖ Where possible, use static lists as opposed to dynamic lists for navigation. When a page is rendered using static lists, it does not have to go back to the server to acquire information, making for faster response time.
- ❖ Use client-side page construction if possible. Many out-of-the-box fragments use client-side mechanisms to build navigation rather than server-side fragments. The client-side fragments will be faster when building site navigation.

See the Site Studio documentation for more details.

PUBLISHING UTILITY/CONNECTION SERVER

The Publishing Utility provides an integrated interface with Site Studio that lets you publish a web site from a content server environment to a pure web server environment. It is based on Connection Server, which runs on a JVM as a separate entity from the Content Server. Because of this, you can tune the Connection Server in the same way you would tune any JVM. See [Java Tuning](#) (page 4-8) for details.

Other methods you can use to tune Publishing Utility performance include:

- ❖ Increase the JVM's maximum heap size. The default is 384MB. Increase this to 512MB at a minimum. To do so, comment out the `JvmCommandLine` setting in the `<install_dir>/bin/intradoc.cfg` file and change the `JAVA_OPTIONS` setting, as in this example:

```
#JvmCommandLine=/home/contribution/shared/os/<os>/j2sdk1.4.1/bin/java -cp
$CLASSPATH $STARTUPCLASS
JAVA_OPTIONS= -Xmx512m
```

See the *Troubleshooting Guide* for more details.

- ❖ On multi-processor systems, increase the number of threads used by the garbage collection facility.
- ❖ Run the Connection Server on its own dedicated machine. This is especially useful for large sites that have frequent publishing needs.
- ❖ Check all trace logs and resolve any 'soft' errors that occur. Soft errors are those that don't stop the Publishing Utility but which may slow down processing.
- ❖ Optimize Site Studio pages for rendering. See [Site Studio](#) (page 5-2) for tips on optimizing Site Studio pages.

INBOUND REFINERY

Several different products are used with the Inbound Refinery. These products can be individually optimized for overall performance improvements.

- ❖ The PDF Converter is an add-on which is used in conjunction with the Inbound Refinery to convert documents to PDF format. As of release 7.0, the JAWS engine is used as the built-in PDF conversion technology. You can set different JAWS options, adjusting them for speed. It's important to find a balance between speed and quality. See the *PDF Converter Administration Guide* for details.
- ❖ Put the Inbound Refinery on a different server than the Content Server, allowing each to have their own resources.
- ❖ Run multiple refineries and make use of *selective queue processing*, which is used to define file types for each refinery. For example, Refinery 1 can be optimized for PowerPoint, Refinery 2 for TIFF conversion, and so on. All the refineries can work simultaneously and one large document will not hold up the queue for documents which follow.

CONTENT PUBLISHER

Content Publisher provides advanced template-based technology to automatically convert and publish business-content forms (for example, those generated in MS Office or Lotus Suite) to fully linked Web sites and pages in HTML, XML, and wireless formats.

Using Content Publisher, you can create projects and either translate those projects manually or automate the translation process. There are several ways to optimize Publisher use:

- ❖ Split up projects into Master and Sub projects. Those areas of the site that need frequent updates can then be scheduled to update as often as needed. Therefore, the Site Server will not need to run as often.
- ❖ After identifying the bottleneck and determining it is not the search or database on the Content Server, try putting different Publishers on different machines. The projects can then work in parallel.
- ❖ Optimize the queries for directory items. See [Publishing Utility/Connection Server](#) (page 5-3) and [Site Studio](#) (page 5-2) for information.
- ❖ Reduce the number of styles and unnecessary elements in the HTML prior to conversion. When the HTML is converted, it will be optimized for use.

CONTENT INTEGRATION/PORTAL SUITE

The Integration Suite provides a scalable integration infrastructure for integrations with Enterprise applications. Content Portal provides reference portlets which can be used immediately or as an example of how to implement portlets in Integration Suite.

The main method of optimizing the Integration and Portal Suites is to utilize caching to its fullest. Service-level caching is available, but the default is to cache only DOC_INFO. You should try to cache data that is public and not tied to users or security (for example, Announcements or News).

Use page-level HTTP caching and application server-specific caching for this purpose.

TOOLS TO ASSIST OPTIMIZATION

Several add-on tools are available to help improve performance and to help you decide what areas to optimize.

Content Tracker

Content Tracker is used to monitor content and server access. It can also be used to perform data extraction to populate a database repository. This data is then viewable with different reporting tools.

With Content Tracker you can determine what content is actively used. This can help you keep your repository small by eliminating unnecessary and unused items.

Using Tracker you can determine how people are searching the repository and optimize search queries based on the information you find.

In addition, Content Tracker information can be sent to another database to perform statistical analysis, freeing up database resources for other use.

Compression

Content Server Compression is used for the check-in of large image files that are automatically converted to a compressed MrSID format. A representation is visible as a thumbnail, which allows consumers to perform high resolution zooming on that image.

Compression of this type reduces the file size of large images, such as maps, satellite photos and x-rays. The image view lowers bandwidth for consumption.

SYSTEM ARCHITECTURE TUNING

OVERVIEW

Performance problems in Content Server can sometimes be caused by underlying system architecture problems. This chapter covers the following aspects of architecture tuning:

- ❖ [General Tips for Performance Improvements](#) (page 6-1)
- ❖ [Common Mistakes](#) (page 6-5)

GENERAL TIPS FOR PERFORMANCE IMPROVEMENTS

There are several checklist items you can use to verify that your architecture is correctly configured to handle the Content Server repository and operations:

- ❖ Always install the most recent upgrades and patches for your operating system and other products. Upgrades can often provide performance improvements without further customization on your part.
- ❖ Check the amount of CPU. The biggest impact on CPU usage comes during searching and web page construction. If you have inadequate CPU resources on the system, performance will suffer.
- ❖ Memory: make sure that there is 1GB per CPU. This number is dependent on the number of Content Server instances on the machine. The Content Server runs on a 32-bit JVM so a single instance cannot use more than 4GB of RAM even though it can use 16 or more CPUs. Whenever possible, let the Java Virtual Machine tune itself unless memory errors occur.

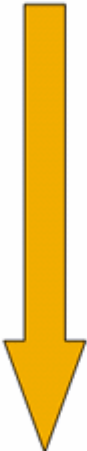
- ❖ Move applications to other machines where the Content Server is not located. The exception to this is the web server, which should remain on the same machine as the Content Server. Where possible, move the Refinery, Publishing, Application Server, and any other applications to other machines.
- ❖ Optimize traffic between the segments of your network. Where possible, keep servers on the same subnet to minimize traffic delays.
- ❖ For large collections, consider using multiple Content Server instances of the collection with Enterprise Search.
- ❖ For systems which have large amounts of consumption, scale your operation horizontally and not vertically. Horizontal scaling involves clustering — setting up multiple nodes and new servers. Vertical scaling involves adding more hardware (CPU and RAM) to existing servers. For sites which have a large amount of contribution, vertical scaling may provide more performance improvements.

Both vertical and horizontal scaling involve adding more hardware. Horizontal scaling may be more effective because JVM thread synchronization may limit the value of vertical scaling in a single instance. It is, however, possible to use vertical scaling with multiple Content Server instances. In either case, the method of scaling is an economic decision.

- ❖ Use proxy servers and reverse proxy servers to optimize network traffic and to distribute CPU and database loads. Proxy servers can decrease page response time by caching copies of frequently-requested pages in one location. When a page request is submitted, the page from the proxy cache is served, rather than passing the request back to the server. See the *Reverse Proxy Resource Guide* and the *Proxy Connections Component Administration Guide* for information about proxy usage.
- ❖ Always optimize before setting up clusters. Make certain that you have tried other performance improvements before investing in the time and money that clustering involves. In some cases, clustering may not solve a problem but may only aggravate it. See the *Clustering Concepts Guide* for details about setting up Content Server in a clustered environment.

Another key factor is the location of system elements. Performance speed slows as elements are located farther apart. The following illustration shows performance improvements based on the location of the system elements.

Figure 6-2 Performance improvements based on system location

Performance	Location		
	Web Server	File System	Database
Fastest  Slowest	Local	Local	Local
	Local	Local	Remote
	Local	SAN	Remote
	Local	NAS	Remote
	Local	Remote	Remote
	Remote	Local	Remote
	Remote	SAN	Remote
	Remote	NAS	Remote
	Remote	Remote	Remote

As you can see, the more distance between key elements, the slower the performance time will be.

Alleviating Storage Problems

As the amount of content in your Content Server instance grows, the vault, weblayout, and search directories will become the largest users of disk space. This can sometimes lead to a shortage of available space on the drive where the Content Server was originally installed.

Moving any or all of these directories to another drive with more space can help alleviate storage problems. While you can use a non-local drive (such as a network drive), this can have a performance impact. The Content Server requires uninterrupted access to all files on any non-local drive. Any latency in accessing these files, especially in the search

directory, will have a direct negative impact on Content Server performance. Therefore, you should use a local drive whenever possible.

The following steps detail how to move these directories. Before changing any of the files mentioned here, be sure to back up your configuration files. You can use any text editor to modify these files.



Important: If you are moving these directories to a network share such as a NAS drive, the user who performs the actions must have full read and write access to the network drive. Otherwise the Content Server will not be able to access the files and files will be unavailable to your users.

Change the Location of the Vault Folder

1. Stop your Content Server.
2. Move the Vault directory to its new location. For example, *d:/vault/*.
3. Add the setting `VaultDir=d:/vault/` to the following configuration files:
 - `<install_dir>/bin/intradoc.cfg`
 - `<refinery_dir>/connections/main/intradoc.cfg` (This file only applies if you are using the Inbound Refinery)
4. Restart your Admin Service, Content Server instance and Inbound Refinery (if applicable).

Change the Location of the Weblayout Folder

1. Stop the Content Server.
2. Move the Weblayout directory to its new location (for example, *d:/weblayout/*).
3. In your web server (IIS, Apache, Sun One) for your Content Server web site, change the Local Path in the Virtual Directory tab to the new weblayout location.
4. Restart your web server.
5. Add the setting `WeblayoutDir=d:/weblayout/` to the following configuration files:
 - `<install_dir>/bin/intradoc.cfg`
 - `<install_dir>/admin/bin/intradoc.cfg`
 - `<refinery_dir>/connections/main/intradoc.cfg` (This file only applies if you are using the Inbound Refinery)

6. Restart your Admin Service, Content Server instance and Inbound Refinery (if applicable).

Change the Location of your Search Index

1. Stop the Content Server.
2. Move the Search directory to its new location (for example, *d:/search/*).
3. Add the setting `SearchDir=d:/search/` to the following configuration file:
 - `<install_dir>/bin/intradoc.cfg`
4. Restart your Content Server instance.

COMMON MISTAKES

The following list describes common mistakes made when setting up infrastructure:

- ❖ Unnecessary firewalls and virus scans. Examine the firewall usage and eliminate or downgrade those that aren't critical to system security.
- ❖ Overuse of one server. If too many applications are running on one server, response time will be slowed for all applications. Examine your server resources and adjust them accordingly.
- ❖ Network architectures have unnecessary layers, such as routers, hubs, and switches. It's important to keep all applications and servers on the same subnet.
- ❖ Inappropriate timing of backups and archives. Make certain that all activities which can cause a heavy system load are performed at off-peak times, when few users are on the system.
- ❖ Inadequate pipeline between server and file system. The Content Server requires more speed between the content and the file system, while many system setups are focused on speed between the server and the database. Make sure that the pipeline between the server and the file system is appropriate.



THIRD PARTY LICENSES

OVERVIEW

This appendix includes a description of the Third Party Licenses for all the third party products included with this product.

- ❖ [Apache Software License](#) (page B-1)
- ❖ [W3C® Software Notice and License](#) (page B-2)
- ❖ [Zlib License](#) (page B-3)
- ❖ [General BSD License](#) (page B-4)
- ❖ [General MIT License](#) (page B-5)
- ❖ [Unicode License](#) (page B-5)
- ❖ [Miscellaneous Attributions](#) (page B-7)

APACHE SOFTWARE LICENSE

- * Copyright 1999-2004 The Apache Software Foundation.
- * Licensed under the Apache License, Version 2.0 (the "License");
- * you may not use this file except in compliance with the License.
- * You may obtain a copy of the License at
- * <http://www.apache.org/licenses/LICENSE-2.0>
- *

- * Unless required by applicable law or agreed to in writing, software
- * distributed under the License is distributed on an "AS IS" BASIS,
- * WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
- * See the License for the specific language governing permissions and
- * limitations under the License.

W3C® SOFTWARE NOTICE AND LICENSE

- * Copyright © 1994-2000 World Wide Web Consortium,
- * (Massachusetts Institute of Technology, Institut National de
- * Recherche en Informatique et en Automatique, Keio University).
- * All Rights Reserved. <http://www.w3.org/Consortium/Legal/>
- *
- * This W3C work (including software, documents, or other related items) is
- * being provided by the copyright holders under the following license. By
- * obtaining, using and/or copying this work, you (the licensee) agree that
- * you have read, understood, and will comply with the following terms and
- * conditions:
- *
- * Permission to use, copy, modify, and distribute this software and its
- * documentation, with or without modification, for any purpose and without
- * fee or royalty is hereby granted, provided that you include the following
- * on ALL copies of the software and documentation or portions thereof,
- * including modifications, that you make:
- *
- * 1. The full text of this NOTICE in a location viewable to users of the
- * redistributed or derivative work.
- *
- * 2. Any pre-existing intellectual property disclaimers, notices, or terms
- * and conditions. If none exist, a short notice of the following form
- * (hypertext is preferred, text is permitted) should be used within the
- * body of any redistributed or derivative code: "Copyright ©
- * [\$date-of-software] World Wide Web Consortium, (Massachusetts

* Institute of Technology, Institut National de Recherche en
* Informatique et en Automatique, Keio University). All Rights
* Reserved. <http://www.w3.org/Consortium/Legal/>
*
* 3. Notice of any changes or modifications to the W3C files, including the
* date changes were made. (We recommend you provide URIs to the location
* from which the code is derived.)
*
* THIS SOFTWARE AND DOCUMENTATION IS PROVIDED "AS IS," AND COPYRIGHT HOLDERS
* MAKE NO REPRESENTATIONS OR WARRANTIES, EXPRESS OR IMPLIED, INCLUDING BUT
* NOT LIMITED TO, WARRANTIES OF MERCHANTABILITY OR FITNESS FOR ANY PARTICULAR
* PURPOSE OR THAT THE USE OF THE SOFTWARE OR DOCUMENTATION WILL NOT INFRINGE
* ANY THIRD PARTY PATENTS, COPYRIGHTS, TRADEMARKS OR OTHER RIGHTS.
*
* COPYRIGHT HOLDERS WILL NOT BE LIABLE FOR ANY DIRECT, INDIRECT, SPECIAL OR
* CONSEQUENTIAL DAMAGES ARISING OUT OF ANY USE OF THE SOFTWARE OR
* DOCUMENTATION.
*
* The name and trademarks of copyright holders may NOT be used in advertising
* or publicity pertaining to the software without specific, written prior
* permission. Title to copyright in this software and any associated
* documentation will at all times remain with copyright holders.
*

ZLIB LICENSE

* zlib.h -- interface of the 'zlib' general purpose compression library
version 1.2.3, July 18th, 2005

Copyright (C) 1995-2005 Jean-loup Gailly and Mark Adler

This software is provided 'as-is', without any express or implied
warranty. In no event will the authors be held liable for any damages

arising from the use of this software.

Permission is granted to anyone to use this software for any purpose, including commercial applications, and to alter it and redistribute it freely, subject to the following restrictions:

1. The origin of this software must not be misrepresented; you must not claim that you wrote the original software. If you use this software in a product, an acknowledgment in the product documentation would be appreciated but is not required.
2. Altered source versions must be plainly marked as such, and must not be misrepresented as being the original software.
3. This notice may not be removed or altered from any source distribution.

Jean-loup Gailly jloup@gzip.org

Mark Adler madler@alumni.caltech.edu

GENERAL BSD LICENSE

Copyright (c) 1998, Regents of the University of California

All rights reserved.

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

"Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.

"Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.

"Neither the name of the <ORGANIZATION> nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.

THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT OWNER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

GENERAL MIT LICENSE

Copyright (c) 1998, Regents of the Massachusetts Institute of Technology

Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:

The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

UNICODE LICENSE

UNICODE, INC. LICENSE AGREEMENT - DATA FILES AND SOFTWARE

Unicode Data Files include all data files under the directories <http://www.unicode.org/Public/>, <http://www.unicode.org/reports/>, and <http://www.unicode.org/cldr/data/>. Unicode Software includes any source code published in the Unicode Standard or under the directories <http://www.unicode.org/Public/>, <http://www.unicode.org/reports/>, and <http://www.unicode.org/cldr/data/>.

Third Party Licenses

NOTICE TO USER: Carefully read the following legal agreement. BY DOWNLOADING, INSTALLING, COPYING OR OTHERWISE USING UNICODE INC.'S DATA FILES ("DATA FILES"), AND/OR SOFTWARE ("SOFTWARE"), YOU UNEQUIVOCALLY ACCEPT, AND AGREE TO BE BOUND BY, ALL OF THE TERMS AND CONDITIONS OF THIS AGREEMENT. IF YOU DO NOT AGREE, DO NOT DOWNLOAD, INSTALL, COPY, DISTRIBUTE OR USE THE DATA FILES OR SOFTWARE.

COPYRIGHT AND PERMISSION NOTICE

Copyright © 1991-2006 Unicode, Inc. All rights reserved. Distributed under the Terms of Use in <http://www.unicode.org/copyright.html>.

Permission is hereby granted, free of charge, to any person obtaining a copy of the Unicode data files and any associated documentation (the "Data Files") or Unicode software and any associated documentation (the "Software") to deal in the Data Files or Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, and/or sell copies of the Data Files or Software, and to permit persons to whom the Data Files or Software are furnished to do so, provided that (a) the above copyright notice(s) and this permission notice appear with all copies of the Data Files or Software, (b) both the above copyright notice(s) and this permission notice appear in associated documentation, and (c) there is clear notice in each modified Data File or in the Software as well as in the documentation associated with the Data File(s) or Software that the data or software has been modified.

THE DATA FILES AND SOFTWARE ARE PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT OF THIRD PARTY RIGHTS. IN NO EVENT SHALL THE COPYRIGHT HOLDER OR HOLDERS INCLUDED IN THIS NOTICE BE LIABLE FOR ANY CLAIM, OR ANY SPECIAL INDIRECT OR CONSEQUENTIAL DAMAGES, OR ANY DAMAGES WHATSOEVER RESULTING FROM LOSS OF USE, DATA OR PROFITS, WHETHER IN AN ACTION OF CONTRACT, NEGLIGENCE OR OTHER TORTIOUS ACTION, ARISING OUT OF OR IN CONNECTION WITH THE USE OR PERFORMANCE OF THE DATA FILES OR SOFTWARE.

Except as contained in this notice, the name of a copyright holder shall not be used in advertising or otherwise to promote the sale, use or other dealings in these Data Files or Software without prior written authorization of the copyright holder.

Unicode and the Unicode logo are trademarks of Unicode, Inc., and may be registered in some jurisdictions. All other trademarks and registered trademarks mentioned herein are the property of their respective owners

MISCELLANEOUS ATTRIBUTIONS

Adobe, Acrobat, and the Acrobat Logo are registered trademarks of Adobe Systems Incorporated.

FAST Instream is a trademark of Fast Search and Transfer ASA.

HP-UX is a registered trademark of Hewlett-Packard Company.

IBM, Informix, and DB2 are registered trademarks of IBM Corporation.

Jaws PDF Library is a registered trademark of Global Graphics Software Ltd.

Kofax is a registered trademark, and Ascent and Ascent Capture are trademarks of Kofax Image Products.

Linux is a registered trademark of Linus Torvalds.

Mac is a registered trademark, and Safari is a trademark of Apple Computer, Inc.

Microsoft, Windows, and Internet Explorer are registered trademarks of Microsoft Corporation.

MrSID is property of LizardTech, Inc. It is protected by U.S. Patent No. 5,710,835. Foreign Patents Pending.

Oracle is a registered trademark of Oracle Corporation.

Portions Copyright © 1994-1997 LEAD Technologies, Inc. All rights reserved.

Portions Copyright © 1990-1998 Handmade Software, Inc. All rights reserved.

Portions Copyright © 1988, 1997 Aladdin Enterprises. All rights reserved.

Portions Copyright © 1997 Soft Horizons. All rights reserved.

Portions Copyright © 1995-1999 LizardTech, Inc. All rights reserved.

Red Hat is a registered trademark of Red Hat, Inc.

Sun is a registered trademark, and Sun ONE, Solaris, iPlanet and Java are trademarks of Sun Microsystems, Inc.

Sybase is a registered trademark of Sybase, Inc.

UNIX is a registered trademark of The Open Group.

Verity is a registered trademark of Autonomy Corporation plc



A

- add-on components
 - tuning
 - batchloader, 5-1
 - connection server, 5-3
 - content integration, 5-5
 - Content Publisher, 5-4
 - dynamic converter, 5-2
 - inbound refinery, 5-4
 - PDF converter, 5-4
 - portal suite, 5-5
 - publishing utility, 5-3
 - Site Studio, 5-2
- additional documentation, 1-2
- adjunct tuning, 1-2
- analysis pitfalls, 3-11
- analyzing benchmarks, 3-7
- archiving usage, 3-3
- auto-update indexer cycle, 5-1

B

- backups
 - timing, 6-5
- bad-case benchmark scenarios, 3-6
- batchloader tuning, 5-1
- batchloading usage, 3-3
- batchloads and database use, 5-2
- benchmarks
 - analysis, 3-7
 - analysis pitfalls, 3-11
 - bad-case scenario, 3-6
 - best-case scenario, 3-6
 - design, 3-1
 - design points, 3-2
 - overview, 3-1
 - results
 - core engine/JVM, 3-9
 - database problems, 3-8
 - file system, 3-10

- index, 3-7
- networks, 3-9
- search, 3-7
- web security filter, 3-10
- running, 3-6
- script design, 3-3
- session setup, 3-3
- setup information, 3-2
- tools, 3-4
- worst-case scenarios, 3-6
- best-case benchmarks, 3-6

C

- caching
 - importance, 2-2
- client-side page construction, 5-3
- clusters, 6-2
- common infrastructure mistakes, 6-5
- Compression, 5-5
- configuration flags, 4-1
- configuration slowdowns, 2-2
- connection server tuning, 5-3
- Content Integration Suite tuning, 5-5
- Content Publisher
 - master and sub projects, 5-4
- Content Publisher tuning, 5-4
- Content Server capacity, 2-3
- Content Server logs, 3-4
- Content Server tracing logs, 3-4
- Content Server tuning, 1-1
- Content Tracker, 5-5
- core engine problem areas, 3-9

D

- database backups, 4-7
- database maintenance
 - backups, 4-7
 - database integrity, 4-7
 - index reorganization, 4-8

- memory, 4-7
- notification, 4-8
- operations, 4-7
- other considerations, 4-7
- overview, 4-6
- plan dependency list, 4-8
- space considerations, 4-7
- space fragmentation, 4-7
- table reorganization, 4-7
- tablespace, 4-7
- database plan notifications, 4-8
- database search, 4-9
- database slowdowns, 3-8
- database tuning, 4-5
- database variables, 4-2
- document audience, 1-2
- document conventions, 1-3
- document overview, 1-2
- Dynamic Converter, 4-9
- dynamic converter tuning, 5-2
- dynamic lists
 - tuning, 5-2

E

- external system tuning, 1-1

F

- file system problem areas, 3-10
- file system size, 2-2
- firewalls, 6-5
- full-text vs. metadata searching, 4-9

G

- general performance improvement, 6-1

H

- horizontal scaling, 6-2

I

- Inbound Refinery
 - selective queue processing, 5-4
- Inbound Refinery tuning, 5-4
- index problem areas, 3-8
- index tuning, 4-2
- indexer variables, 4-2
- infrastructure

- common mistakes, 6-5
- locations, 6-2
- subnets, 6-2
- tuning, 6-1

J

- Java tuning, 4-8
- JVM garbage collection tuning, 4-8
- JVM heap size, 5-3
- JVM problem areas, 3-9
- JVM tuning, 4-8

L

- log files, 3-4
- logging error slowdowns, 3-2

M

- memory maintenance
 - databases, 4-7
- memory size, 2-2
- moving
 - search directory, 6-3
 - vault directory, 6-3
 - weblayout directory, 6-3

N

- network problem areas, 3-9

O

- optimization tools
 - compression, 5-5
 - Content Tracker, 5-5
- other tuning variables, 4-2
- overview, 1-1

P

- PDF converter
 - tuning, 5-4
- performance improvement tips, 6-1
- performance speed and infrastructure, 6-2
- plan dependency list, 4-8
- Portal Suite tuning, 5-5
- processor type, 2-2
- projects

- master and sub, 5-4
- publishing usage, 3-3
- publishing utility tuning, 5-3

R

- related documentation, 1-2
- resource checks, 3-6
- reverse proxy usage, 6-2

S

- search collection optimization, 5-2
- search problem areas, 3-8
- search variables, 4-2
- security and groups, 2-3
- security considerations, 4-10
- selective queue processing, 5-4
- server overuse, 6-5
- server pipeline, 6-5
- service optimization, 5-3
- Site Studio page simplification, 4-9
- Site Studio tuning, 5-2
- space considerations
 - databases, 4-7
- space fragmentation, 4-7
- static lists vs. dynamic lists, 5-3
- Stellent Content Publisher, 4-9
- storage problems, 6-3
- subnets, 6-5
- system architecture tuning, 6-1
- system slowdowns
 - caching, 2-2
 - causes, 2-2
 - configurations, 2-2
 - file system size, 2-2
 - hardware, 2-2
 - logging errors, 3-2
 - memory size, 2-2
 - obvious signs, 2-1
 - processor type, 2-2
 - questions, 2-3
 - security and groups, 2-3
 - typical causes, 2-2

T

- table reorganization, 4-7
- Template Selection Rules, 5-2
- tracing options, 3-4
- tracing reports, 3-4
- tuning

- add-on components, 5-1
 - batchloader, 5-1
 - connection server, 5-3
 - Content Integration Suite, 5-5
 - Content Publisher, 5-4
 - dynamic converter, 5-2
 - Inbound Refinery, 5-4
 - Portal Suite, 5-5
 - publishing utility, 5-3
 - Site Studio, 5-2
- adjunct tuning, 1-2
- available log files, 3-4
- backups, 6-5
- basic steps, 2-4
- clusters, 6-2
- configuration flags, 4-1
- Content Server, 1-1
- CPU, 6-1
- database tuning, 4-5
- database variables, 4-2
- Enterprise Search, 6-2
- external system tuning, 1-1
- firewalls, 6-5
- horizontal vs. vertical scaling, 6-2
- Idoc Script caching, 4-5
- index tuning, 4-2
- indexer variables, 4-2
- Java tuning, 4-8
- machine usage, 6-2
- memory, 6-1
- network traffic, 6-2
- other options, 4-9
- other variables, 4-2
- overview, 2-1
- proxy servers, 6-2
- rules, 2-2
- search variables, 4-2
- simplify Site Studio pages, 4-9
- steps, 2-2
- tips, 6-1
- types, 1-1
- virus scanners, 6-5
- Web Server Filter, 4-10

U

- usage questions, 3-3
- user types, 3-3

V

- vertical scaling, 6-2

Index

virus scanners, 3-6
virus scans, 6-5

W

web security file problems, 3-10
Web Server Filter, 4-10
worst-case benchmark scenarios, 3-6