

Domino for iSeries Sizing and Performance Tuning

Correctly sizing a server with the Workload Estimator

Tips and techniques for tuning Domino for iSeries

Information on iSeries Dedicated Server for Domino



Wilfried Blankertz
Christina Fasth
Kim Greene
Deb Landon
Brandon Rau
Colin Stamp



International Technical Support Organization

Domino for iSeries Sizing and Performance Tuning

April 2002

Take Note! Before using this information and the product it supports, be sure to read the general information in “Special notices” on page xi.

Second Edition (April 2002)

This edition applies to Release 5.0.8 of Domino for iSeries for use with OS/400 Version 5 Release 1 (V5R1) and later.

The first edition of this redbook was published in October 1999 and was originally entitled *Lotus Domino for AS/400: Performance, Tuning, and Capacity Planning*. It applied to Release 4.6.3 of Domino for AS/400 for use with OS/400 Version 4 Release 2 and later.

Comments may be addressed to:

IBM Corporation, International Technical Support Organization
Dept. JLU Building 107-2
3605 Highway 52N
Rochester, Minnesota 55901-7829

When you send information to IBM, you grant IBM a non-exclusive right to use or distribute the information in any way it believes appropriate without incurring any obligation to you.

© Copyright International Business Machines Corporation 1999, 2002. All rights reserved.

Note to U.S. Government Users - Documentation related to restricted rights - Use, duplication or disclosure is subject to restrictions set forth in GSA ADP Schedule Contract with IBM Corp.

Contents

Special notices	xi
IBM trademarks	xii
Preface	xv
The team that wrote this edition of the Redbook	xv
The team that wrote the first edition of this redbook	xvii
Special notice	xviii
Comments welcome	xviii
Summary of changes	xix
Second Edition, April 2002	xix
Chapter 1. Introduction	1
1.1 Conclusions and references	2
Chapter 2. Overview of Lotus Domino for iSeries	3
2.1 Why use Domino for iSeries	4
2.1.1 Scalability	4
2.1.2 Reliability and availability	4
2.1.3 Integration	4
2.1.4 Proven security	5
2.2 Planning a Domino for iSeries server	5
2.2.1 Developing a Notes naming structure	5
2.2.2 Choosing the right replication topology	6
2.2.3 How security relates	7
2.2.4 What remains	7
2.3 The value of sizing, performance analysis, and tuning	8
2.3.1 Audience	9
2.4 Domino architecture	9
2.4.1 Layer 0: The kernel	10
2.4.2 Layer 1: Kernel extensions	10
2.4.3 Layer 2: System and core applications	11
2.4.4 Layer 3: Layered end-user applications	11
2.5 Tools available	11
2.5.1 Collection Services	12
2.5.2 Performance Management/400	12
2.5.3 Performance Tools (5722-PT1)	12
2.5.4 iSeries Performance Explorer (PEX)	13
2.5.5 Management Central	13
2.5.6 BEST/1	13
2.5.7 Server.Load	14
2.5.8 Performance Navigator	14
2.5.9 NotesBench	14
2.5.10 GroupSizr	15
2.5.11 WEBSizr	15
2.5.12 LoadRunner	15
2.5.13 Tivoli Manager for Domino	16
2.5.14 Tivoli Application Response Measurement (ARM)	16

2.6 NSD for OS/400 (Dump utility)	17
2.6.1 Activating the NSD tool	17
Chapter 3. Sizing Domino for iSeries using the Workload Estimator	19
3.1 Introduction to Workload Estimator	20
3.2 Benchmarks to determine the load of specific workloads	21
3.3 Workload Estimator terms	23
3.3.1 Workload	23
3.3.2 Instance	24
3.4 Domino sizing concepts in the Workload Estimator	24
3.4.1 Concurrent users	24
3.4.2 Number of partitions	26
3.4.3 Clustering	27
3.5 Domino mail concepts	28
3.5.1 Mail access types	28
3.6 Domino application concepts in Workload Estimator	30
3.6.1 Example application comparisons	31
3.6.2 Database capacity	34
3.7 Characterizing existing applications	35
3.8 Additional options for more accurate estimation	36
3.9 Using IBM Workload Estimator for iSeries results	38
3.10 Additional tips and insights	44
3.11 Consolidating Domino servers from other platforms	46
3.11.1 Capabilities of various iSeries and AS/400 models	48
3.12 Additional resources	51
Chapter 4. Basic concepts of performance analysis	53
4.1 Defining a Domino for iSeries performance methodology	54
4.1.1 Relationship between Domino and the iSeries server	55
4.2 Understanding Domino on the iSeries server	56
4.2.1 Transactions in Domino	56
4.2.2 Domino base functions	56
4.2.3 Domino advanced functions	57
4.3 Domino tasks on the iSeries server	57
4.4 Application development performance	58
4.5 Lotus Domino overall performance on the iSeries server	59
4.6 The Queuing Multiplier (QM) curve	60
4.7 Dividing the CPU utilization	61
4.8 A brief discussion of threads	62
4.8.1 Threads on the iSeries server	63
4.8.2 Other Domino task that use threads	64
4.9 A brief discussion on run attributes	64
4.9.1 Run priority	64
4.9.2 How a change in run priority affects a job	66
4.9.3 Job priority considerations	66
4.9.4 Time slice parameter and tuning	66
4.9.5 How the system manages job and run priorities	67
4.10 Reviewing iSeries performance using CL commands	67
4.10.1 Work with System Status (WRKSYSSTS) command	68
4.10.2 Information about thread state transitions	69
4.10.3 Work with Active Jobs (WRKACTJOB) command	70
4.10.4 Work with System Activity (WRKSYSACT) command	71
4.10.5 Work with Disk Status (WRKDSKSTS) command	73

4.10.6	Observing network performance	74
4.10.7	Work with System Values (WRKSYSVAL) command	79
4.10.8	Display System Log (DSPLOG) command	81
4.10.9	Start System Service Tools (STRSST) command	82
4.11	Collecting performance data	83
4.11.1	Starting Collection Services	83
4.11.2	Using Work with System Activity (WRKSYSACT) with an output file	89
4.12	Management Central performance monitor	89
4.12.1	Monitoring real-time system performance	90
4.12.2	Viewing graph history of monitor data	102
4.12.3	Exporting Graph History to PC files for analysis and printing	103
4.12.4	Combining Management Central's CPU Utilization metrics	104
4.12.5	Job monitoring	105
4.13	Reviewing data using Performance Tools/400	114
4.13.1	Print System Report (PRTSYSRPT) command	114
4.13.2	Print Component Report (PRTCPTRPT) command	114
4.13.3	Print Transaction Report (PRTTNSRPT) command	114
4.13.4	Print Activity Report (PRTACTRPT) command	115
4.13.5	Convert Performance Thread Data (CVTPFRTHD) command	116
4.14	Performance Explorer (PEX)	116
4.14.1	Add PEX Definition (ADDPEXDFN) command	116
4.14.2	Start PEX Session (STRPEX) command	116
4.14.3	End PEX Session (ENDPEX) command	117
4.14.4	Print PEX Report (PRTPEXRPT) command	117
4.14.5	Performance Explorer trace points for Domino	117
4.14.6	TPROF PEX trace	123
Chapter 5.	Domino logs and statistics	127
5.1	The commands	128
5.1.1	The show tasks command	128
5.1.2	Understanding the Domino statistics	132
5.1.3	Show Domino open database statistics (sh DBS)	136
5.1.4	Platform-dependent statistics	137
5.1.5	Show Domino Transactions	143
5.2	The reports	145
5.2.1	The Log database (log.nsf)	146
5.2.2	The Statistics and Reporting database (statrep.nsf)	149
5.3	Combining performance data from Domino and iSeries	152
5.4	Monitoring the Notes Remote Procedure Calls (NRPC)	156
5.4.1	Using a network sniffer	156
5.4.2	Lotus Client NRPC Monitoring Tool Client_Clock	157
Chapter 6.	Tuning the iSeries server for Lotus Domino	161
6.1	Tuning the iSeries server	162
6.2	Automatic performance adjustment (QPFRADJ)	163
6.2.1	The interactive iSeries tuning chart	164
6.2.2	Summary	165
6.3	Determining Domino workload	166
6.4	Choosing which memory pool to use	166
6.4.1	Using expert cache	169
6.4.2	Activity levels of memory pools	170
6.4.3	Faulting rates	171
6.5	Choosing which processor priority to use	173

6.5.1 Changing the run priority of Domino tasks	174
6.6 DASD: System ASP, user ASPs, and independent ASPs	175
6.7 Unleashing iSeries work management on Domino servers.	176
6.7.1 Configuring a Domino server	176
6.7.2 Starting a Domino server	176
6.7.3 Working with Domino jobs.	177
6.8 Online backup with BRMS	178
6.9 Network tuning	179
6.9.1 Maximum Transmission Unit (MTU) Size	180
6.9.2 TCP/IP buffer size.	180
6.9.3 Port filtering.	180
6.9.4 Duplex.	181
6.9.5 TCPONLY for Ethernet	181
Chapter 7. Tuning Lotus Domino for better performance on iSeries	183
7.1 Basics of Domino tuning	184
7.2 Relating Domino tasks to iSeries jobs.	184
7.3 Domino parameters that affect performance.	185
7.4 Domino Database indexing: Controlling the UPDATE task.	185
7.4.1 Components of the Domino database indexer	185
7.4.2 UPDATE task CPU utilization in a normal environment	188
7.4.3 The UPDATE task and CPU utilization in a problem environment	188
7.4.4 Affecting the UPDATE task behavior through the view properties	190
7.4.5 Detecting UPDATE task problems	192
7.4.6 Making the changes	196
7.4.7 Conclusions	199
7.5 TCP/IP between Domino servers on the same iSeries	199
7.6 GTR search engine version 3.4	199
7.7 Using the Extended Directory Catalog	199
7.7.1 Comparing EDC to a standard directory catalog (DirCat).	199
7.7.2 Using Extended Directory Catalog to improve performance.	200
7.8 Investigating mail backlogs and performance issues	200
7.9 Domino console logging	200
7.10 Number of users per Domino server.	201
7.11 Using a separate partition for the R5 SMTP MTA server	202
7.12 Excessive translations on iSeries and zSeries	202
7.12.1 Opening up any suspect job using the new Domino panel group.	202
7.13 Optimizing mail: White space, compact, and quotas	205
7.13.1 Mail file size	209
7.14 Domino memory management	210
7.14.1 NSF_Buffer_Pool_Size or Nsf_Buffer_Pool_Size_mb	211
7.14.2 How queues relate to the NSF_Buffer_Pool_Size	219
7.15 Controlling the details of Domino logging	223
7.16 iSeries environment variable settings	224
7.17 Domino for iSeries specific notes.ini parameters	224
7.18 CORBA and Java servlets	227
Chapter 8. Understanding the Domino server jobs	229
8.1 Domino server jobs	230
8.2 Domino server jobs (always necessary)	232
8.2.1 ADMINP: Administration process	232
8.2.2 AMGR: Agent manager.	234
8.2.3 QNNINSTS: The 'Watchdog'.	239

8.2.4	ROUTER: The mail router.	240
8.2.5	SERVER: The main server.	243
8.2.6	STATLOG: Database activity logger.	244
8.2.7	UPDATE: The Indexer task.	244
8.3	Domino server jobs (often used).	245
8.3.1	CLADMIN: Cluster administration process.	245
8.3.2	CLDBDIR: Cluster Database Directory Manager.	246
8.3.3	CLREPL: Cluster Replicator.	246
8.3.4	COLLECT: Statistics collector.	246
8.3.5	EVENT.	247
8.3.6	HTTP: The Web server.	249
8.3.7	LOGASIO: Log asynchronous I/O process.	249
8.3.8	REPLICA: The replication task.	249
8.3.9	SCHED: Schedule Manager.	252
8.3.10	SMTP: Internet Mail Router.	253
8.4	Domino server jobs (very likely not used).	253
8.4.1	BILLING.	253
8.4.2	CALCONN: Calendar connector.	254
8.4.3	QNNINADD: Directory synchronization add-in task.	255
8.4.4	REPORT: Statistic reporter.	256
8.4.5	STATS: Statistics on demand via e-mail.	256
8.5	Domino server database housekeeping jobs.	256
8.5.1	COMPACT.	256
8.5.2	UPDALL.	257
8.5.3	Fixup.	257
8.6	Series SMTP: AnyMail/400 Mail Server Framework (MSF).	258
8.7	Domino server QNNxxxx programs and QSYSWRK.	258
Chapter 9. Integration with DB2 performance tips and techniques.		259
9.1	Choosing a storage container.	260
9.1.1	Domino as a storage container.	260
9.1.2	DB2 UDB as a data storage container.	261
9.1.3	Why Domino does not scale like DB2 UDB.	262
9.2	Integrating Domino and DB2 UDB.	263
9.2.1	Integration options.	263
9.2.2	Connection pooling.	271
9.3	Questions to consider when integrating Domino and DB2.	272
Chapter 10. Clustering and partitioning.		273
10.1	Domino partitioning.	274
10.1.1	Why partition Domino on iSeries servers.	274
10.1.2	How Domino partitioning works.	275
10.1.3	Planning for Domino partitioning.	276
10.1.4	Partitioning tips.	277
10.1.5	Communication between Domino servers on the same iSeries server.	278
10.2	Clustering.	278
10.2.1	iSeries clustering.	279
10.2.2	ClusterProven Domino for iSeries.	280
10.2.3	Domino clustering.	283
10.2.4	Why cluster Domino servers.	283
10.2.5	How a Domino cluster works.	284
10.2.6	ICM and the Domino HTTP Clustering components and technology.	287
10.2.7	Planning for Domino clustering.	289

10.2.8	Domino clustering limitations and recommendations	290
10.2.9	Mail database integration for high availability	291
10.2.10	Remote access with iNotes.	292
10.2.11	Application hub server	292
10.2.12	Network performance	292
10.2.13	Server_Availability_Index and Server_Availability_Threshold	293
10.2.14	Tools	293
10.2.15	Server-based tools	295
10.2.16	Clustering tips	297
Chapter 11.	Internet and intranet performance tips.	301
11.1	Internet and intranet protocols	302
11.2	HTTP performance tuning.	302
11.2.1	HTTP terminology	304
11.2.2	Optimizing HTTP threads	305
11.2.3	Optimizing HTTP cache settings.	308
11.2.4	Optimizing HTTP logging for performance	310
11.2.5	Best practices for Domino HTTP server	310
11.2.6	Summary of HTTP issues, architecture, and tuning possibilities	315
11.2.7	Performance tuning of time-out values on the Domino Web server	315
11.2.8	Full Text Search information in a document over the Web	317
11.3	Structure of the SMTP MTA	317
11.3.1	SMTP performance improvements in Domino R5.	319
11.3.2	Best practices for SMTP messaging using partitioning.	323
11.4	Other Internet and intranet protocols and functions	327
11.4.1	IMAP or POP3 on Domino Server within a cluster and failover	327
11.4.2	IMAP server	327
11.4.3	POP3 server	329
11.4.4	LDAP server	330
11.4.5	NNTP server	334
11.4.6	iNotes	337
Chapter 12.	The iSeries Dedicated Server for Domino	341
12.1	Dedicated Server for Domino	342
12.2	Dedicated Server performance behavior and capacity	342
12.2.1	Enhancements with OS/400 V5R1	343
12.2.2	Dedicated Server for Domino workload environments	345
12.2.3	Dedicated server capacity.	348
12.3	Evaluating iSeries application integration on Dedicated Server for Domino	350
12.3.1	DB2 database processing defined	351
12.3.2	Accessing external databases on the same iSeries server.	354
12.3.3	Accessing external databases on a different iSeries server	355
12.3.4	Java and Domino integration	356
12.3.5	WebSphere and Domino integration.	356
12.3.6	File serving on a Dedicated Server.	357
12.3.7	Backup Recovery and Media Services on a Dedicated Server.	358
12.3.8	Logical partitioning (LPAR) on a Dedicated Server.	360
12.3.9	Linux on a Dedicated Server for Domino	361
12.4	Observing Dedicated Server performance	362
12.4.1	Operations Navigator: Management Central.	362
12.5	Observing Domino processing with Performance Explorer.	365
12.6	Sizing a Dedicated Server for your Domino application	368
12.6.1	Special tuning for the Dedicated Server	369

Chapter 13. Domino transaction logging on iSeries	371
13.1 Storage management on iSeries.	372
13.1.1 Single-level storage	372
13.1.2 When data is actually written to disk.	373
13.1.3 Auxiliary storage pools	373
13.2 Domino transaction logging and how it operates	376
13.2.1 Performance improvement due to Domino transaction logging	376
13.3 Performance tests with TxL on Domino for iSeries.	381
13.3.1 Test environment	381
13.3.2 Test results	382
13.3.3 Conclusions	382
13.4 When to use Domino for iSeries transaction logging	384
13.4.1 Incremental online backup with BRMS	384
13.5 Miscellaneous tips for Domino transaction logging	385
13.5.1 What is the Database Instance ID (DBIID)	385
13.5.2 How transaction logging technically works	386
 Appendix A. Capacity planning for Domino server using BEST/1	391
The need for capacity planning	392
Capacity planning prerequisites	392
Basic capacity planning process	392
CPW values versus Domino mail user ratings	393
Capacity planning unit of measure	395
Business transaction	395
Measurement	395
Domino server workload definition	396
What the different workloads are	396
Splitting up the workloads	398
Other Domino workload concerns	398
BEST/1 for Domino server capacity planning	399
Introduction	399
iSeries Domino server modeling	400
Measuring workload and system resource utilization	403
Creating a model using measured data	404
Validating the BEST/1 model	413
Saving the BEST/1 model	423
 Appendix B. SMTP and AnyMail/400 Mail Server Framework (MSF) jobs	427
 Appendix C. Important notes.ini parameters	435
Steps for changing the notes.ini file	436
Description of notes.ini variables	436
 Appendix D. Domino R5 statistics	453
 Related publications	477
IBM Redbooks	477
Other resources	479
Referenced Web sites	480
How to get IBM Redbooks	481
IBM Redbooks collections	481
 Index	483

Special notices

References in this publication to IBM products, programs or services do not imply that IBM intends to make these available in all countries in which IBM operates. Any reference to an IBM product, program, or service is not intended to state or imply that only IBM's product, program, or service may be used. Any functionally equivalent program that does not infringe any of IBM's intellectual property rights may be used instead of the IBM product, program or service.

Information in this book was developed in conjunction with use of the equipment specified, and is limited in application to those specific hardware and software products and levels.

IBM may have patents or pending patent applications covering subject matter in this document. The furnishing of this document does not give you any license to these patents. You can send license inquiries, in writing, to the IBM Director of Licensing, IBM Corporation, North Castle Drive, Armonk, NY 10504-1785.

Licensees of this program who wish to have information about it for the purpose of enabling: (i) the exchange of information between independently created programs and other programs (including this one) and (ii) the mutual use of the information which has been exchanged, should contact IBM Corporation, Dept. 600A, Mail Drop 1329, Somers, NY 10589 USA.



Such information may be available, subject to appropriate terms and conditions, including in some cases, payment of a fee.

The information contained in this document has not been submitted to any formal IBM test and is distributed AS IS. The use of this information or the implementation of any of these techniques is a customer responsibility and depends on the customer's ability to evaluate and integrate them into the customer's operational environment. While each item may have been reviewed by IBM for accuracy in a specific situation, there is no guarantee that the same or similar results will be obtained elsewhere. Customers attempting to adapt these techniques to their own environments do so at their own risk.

Any pointers in this publication to external Web sites are provided for convenience only and do not in any manner serve as an endorsement of these Web sites.

IBM trademarks

The following terms are trademarks of the International Business Machines Corporation in the United States and/or other countries:

e (logo)® 	Redbooks (logo)™ 	AIX®
AS/400®	AS/400e™	Balance®
CICS®	ClusterProven™	DB2®
DB2 Universal Database™	DFS™	Distributed Relational
DRDA®	DXT™	Database Architecture™
IBM®	IMS™	e (logo)®
MQSeries®	Net.Commerce™	iSeries™
Network Station™	OfficeVision®	Netfinity®
OS/2®	OS/400®	OfficeVision/400™
PartnerWorld®	Perform™	PAL®
Redbooks™	SecureWay®	pSeries™
SP1®	Tivoli®	SP™
VisualInfo™	WebSphere®	VisualAge®
zSeries™	Lotus®	xSeries™
cc:Mail®	Lotusphere®	1-2-3®
Notes®	Sametime®	Lotus Notes®
iNotes™	Lotus Discovery Server™	Domino™
Lotus Sametime™	Mobile Notes™	Lotus QuickPlace™
		QuickPlace™

Other company trademarks

The following terms are trademarks of other companies:

Tivoli, Manage. Anything. Anywhere., The Power To Manage., Anything. Anywhere., TME, NetView, Cross-Site, Tivoli Ready, Tivoli Certified, Planet Tivoli, and Tivoli Enterprise are trademarks or registered trademarks of Tivoli Systems Inc., an IBM company, in the United States, other countries, or both. In Denmark, Tivoli is a trademark licensed from Kjøbenhavns Sommer - Tivoli A/S.

C-bus is a trademark of Corollary, Inc. in the United States and/or other countries.

Java and all Java-based trademarks and logos are trademarks or registered trademarks of Sun Microsystems, Inc. in the United States and/or other countries.

Microsoft, Windows, Windows NT, and the Windows logo are trademarks of Microsoft Corporation in the United States and/or other countries.

PC Direct is a trademark of Ziff Communications Company in the United States and/or other countries and is used by IBM Corporation under license.

ActionMedia, LANDesk, MMX, Pentium and ProShare are trademarks of Intel Corporation in the United States and/or other countries.

UNIX is a registered trademark in the United States and other countries licensed exclusively through The Open Group.

SET, SET Secure Electronic Transaction, and the SET Logo are trademarks owned by SET Secure Electronic Transaction LLC.

Other company, product, and service names may be trademarks or service marks of others.

Preface

Explore the methodologies and approaches to assist in providing optimal performance of Lotus Domino for iSeries! This IBM Redbook targets technical professionals who are responsible for, or must advise on, the installation and administration of Lotus Domino servers on IBM @server iSeries servers. It can also be used by performance specialists to gain knowledge on how to collect, measure, analyze, and extrapolate performance data from Domino servers. This redbook offers you the ability to translate these methodologies into your own situation and come up with specific performance analysis strategies and a set of tuning parameters for the best possible performance.

This redbook shows a detailed approach for:

- ▶ Estimating an appropriate configuration for a new iSeries server to run Domino
- ▶ Measuring Domino application performance
- ▶ Tuning Domino servers and OS/400 resources for optimal use
- ▶ Understanding the impact of many configuration settings
- ▶ Considering partitioning, clustering, the use of text search, indexing, and views
- ▶ Improving throughput of the integration of Domino and DB2 UDB for iSeries
- ▶ Analyzing the impact of the various Domino server tasks
- ▶ Tuning Domino HTTP server workloads

Note: This redbook reflects the IBM @server iSeries server name. Throughout this redbook, we use the shortened version “iSeries” to refer to both AS/400e and iSeries servers. Although the examples in this redbook were conducted using the iSeries running OS/400 V5R1, they should work the same way on AS/400e servers running OS/400 V5R1.

The team that wrote this edition of the Redbook

This second edition of the Redbook was produced by a team of specialists from around the world working at the International Technical Support Organization, Rochester Center.



Wilfried Blankertz is an Technical Support Specialist for iSeries in IBM EMEA region Central located in Frankfurt/Germany. From 1995 to 1998, he was assigned to the International Technical Support Organization, Rochester Center. Here, he wrote extensively and taught IBM classes worldwide on all areas of AS/400 Groupware solutions and Systems Management. Before joining the ITSO, he worked as a systems engineer in IBM Germany supporting customers with the AS/400 system and its predecessor systems (IBM System /3, /32, /34, /36, and /38) for over 26 years. While he still focuses on iSeries technical support, he is also a Certified Lotus Professional for Domino Administration for Domino R5 Application Development and Domino Administration.



Christina Fasth is a Technical Support Specialist for iSeries in IBM Sweden, where she provides operational and software support to customers. Before joining IBM in 1996, she worked with IBM Systems S/3, S/3X, and AS/400 as a Systems Operator, Application Developer, System Engineering, and IT Manager. Her areas of expertise include Domino for iSeries and Client Servers.



Kim Greene, president of Kim Greene Consulting, Inc., specializes in Domino for iSeries consulting, customized education, and performance. She has over four years of experience with Domino and 11 years of experience with the AS/400 and iSeries platform. Kim specializes in iSeries Domino performance analysis, system, and application tuning, enterprise integration with back-end applications and data, and Java and Domino integration. She is also a frequent writer for technology magazines and speaks at many conferences. Previously, she was employed at IBM in Rochester, Minnesota, in the PartnerWorld for Developers (PWD) organization helping IBM Business Partners incorporate Domino into their existing applications. She worked on several areas of OS/400 performance in the AS/400 development laboratory.



Deb Landon is an IT Specialist at the International Technical Support Organization, Rochester Center, focusing on Domino for iSeries. She writes extensively and teaches IBM classes worldwide on Domino for iSeries. Before joining the ITSO in November of 2000, she was a member of the PartnerWorld for developers, iSeries team supporting business partners in the area of Domino for iSeries. She was a member of the original team who created the highly successful Domino Days event that has since been replicated worldwide.



Brandon Rau is a Staff Software Engineer in the Rochester Support Center. He has seven years of experience with AS/400 and the iSeries specializing in the areas of performance and work management. He has written performance articles published in the *iSeries* magazine, *The Link*, and *Rochester Support Center Newsletter*.



Colin Stamp has been with Lotus/IBM for six years, where he works as a Senior Lotus Service Manager (LSM) and a Senior FSS consultant and specialist, in the field resolving performance and capacity issues for both customers and for Lotus Professional Services Consultancy colleagues on ongoing projects. Prior to joining the LSM team as a Senior Messaging, Communications, Platforms, Performance & High Availability specialist, he worked with the Lotus SoftSwitch Team in a Senior Support, Troubleshooter, and Consultancy capacity. Previously, he worked 10 years for Data General as a Senior Communications Support Engineer. Colin has written a number of white papers around Domino (transaction logging, High Availability, Mailfile usage).

Thanks to the following people for their invaluable contributions to this project:

Dave Herbeck
Marcy Howerter
Dave Johnson
Don Morrison
Joseph H. Peterson
Domino for iSeries development

James Cioffi
Michael P Martin
Richard Smitherman
Domino for iSeries Center of Competency

Michael Gilley
Walter Scanlan
Domino for iSeries Support

The team that wrote the first edition of this redbook

The original version of this redbook was produced by a team of specialists from around the world working at the International Technical Support Organization, Rochester Center.

Gottfried Schimunek
Justine Middleton
Collier Johnson
Chris Larsen
Craig Nash
Petri Nuutinen
Rick Underwood
Mark Van Houten

Thanks to the following people for their invaluable contributions to the first edition of this project:

Tom Budnik
Terry O'Brien
Jeff Tenner
IBM Domino for AS/400 Development

Wilfried Blankertz
IBM Germany, AS/400 Technical Support Center

George Gaylord
IBM Chicago, USA

Mike Denney
Kim Greene
Chuck Stupca
IBM Rochester, USA

Special notice

This publication is intended to help IBM Business Partners and IBM Global Service personnel to measure, analyze, and extrapolate performance data from Lotus Domino servers running on the iSeries server and to do tuning to improve performance of the servers. The information in this publication is not intended as the specification of any programming interfaces that are provided by the iSeries server. See the PUBLICATIONS section of the IBM Programming Announcement for iSeries for more information about what publications are considered to be product documentation.

Comments welcome

Your comments are important to us!

We want our IBM Redbooks to be as helpful as possible. Send us your comments about this or other Redbooks in one of the following ways:

- ▶ Use the online **Contact us** review redbook form found at:
ibm.com/redbooks
- ▶ Send your comments in an Internet note to:
redbook@us.ibm.com
- ▶ Mail your comments to the address on page ii.

Summary of changes

This section describes the technical changes made in this edition of the book and in previous editions. This edition may also include minor corrections and editorial changes that are not identified.

Summary of Changes
for SG24-5162-01
for *Domino for iSeries Sizing and Performance Tuning*
as created or updated on April 3, 2002.

Second Edition, April 2002

This revision reflects the addition, deletion, or modification of new and changed information described below.

New information

- ▶ Chapter 3, “Sizing Domino for iSeries using the Workload Estimator” on page 19, is completely new.
- ▶ The contents of the former Chapter 7, “Capacity Planning for Domino Server” have been moved to Appendix A, “Capacity planning for Domino server using BEST/1” on page 391.
- ▶ Chapter 7, “Tuning Lotus Domino for better performance on iSeries” on page 183, now contains the former Section 6.2 with many changes and additions.
- ▶ New paragraphs and sections have been added to all other chapters.

Changed information

- ▶ Many pieces of information have been changed to reflect the differences due to the enhancements of the Domino releases after Domino for AS/400 R4.6 through Domino for iSeries R5.08, as well as OS/400 Version 5 Release 1.



Introduction

Lotus Domino for iSeries has excellent, scalable performance. It has been proven to be a superior platform for server consolidation since its release. It has been the performance leader as measured by the publicly available NotesBench Benchmark. However, users do not typically use the NotesBench workload in their environments. Therefore, some guidance on how to optimize performance in customer environments is required. Integrating Domino with commercial applications and data in relational databases also needs careful consideration when maximizing throughput and providing the best possible response times.

1.1 Conclusions and references

This section describes briefly some of the most important topics covered in this book. It also points you directly to the appropriate place in this redbook for finding more details.

- ▶ What size of an iSeries server do you need to support 1000 Domino users? Unfortunately, we cannot answer this question in a single sentence without knowing more details. See Chapter 3, “Sizing Domino for iSeries using the Workload Estimator” on page 19, to learn about the additional details you need.
- ▶ If you know Lotus Domino very well, but wonder how you can use OS/400 functions on a IBM @server iSeries server to analyze why Domino or other applications are not performing well, read Chapter 4, “Basic concepts of performance analysis” on page 53.

Lotus Domino provides several logs and statistics reports, are described in Chapter 5, “Domino logs and statistics” on page 127. This information can assist you in understanding bottlenecks and the throughput of your server.

- ▶ When you use partitioning to create multiple servers, you are able to allocate different priorities and memory pools to each server. You can find information about memory storage pools in 6.4, “Choosing which memory pool to use” on page 166, and 6.5, “Choosing which processor priority to use” on page 173.
- ▶ Run separate functions in separate Domino partitions.

You may find it much easier to monitor performance and tune effectively if functions, such as mail and applications, are run in different Domino servers or partitions. Partitioning allows you to configure multiple Domino servers on one iSeries server. You can find information about partitioning in 7.10, “Number of users per Domino server” on page 201, and 10.1, “Domino partitioning” on page 274. Do not run server tasks that you do not need.
- ▶ If your installation is not using calendaring and scheduling, it is a waste of resources to run the Calendar Connector (CALCONN) task on the server. Another function to look at is *directory synchronization*. Directory synchronization is automatically turned on when you configure a Domino server on the iSeries. If you will not use this function, choose to not enable the function when you configure your Domino server or change your existing Domino configure to remove its function. You remove tasks by editing the notes.ini file to take them out of the line ServerTasks. See Chapter 8, “Understanding the Domino server jobs” on page 229, to learn about the usage of the most important Domino server tasks.



Overview of Lotus Domino for iSeries

This chapter explains the benefits of running Lotus Domino on an iSeries server. It describes the planning steps that you have to take when implementing Domino. Plus, it outlines some tools that you can use for monitoring performance.

2.1 Why use Domino for iSeries

Domino for iSeries is a powerful solution for building and deploying both business and e-business applications. As a server, the iSeries server provides reliability and scalability, which are essential in the dynamic and often unpredictable e-business environment. Domino for iSeries has the power and tools to enhance existing business applications and reach new levels of collaboration and coordination.

2.1.1 Scalability

Within a single architecture, the iSeries spans a vast performance spectrum. On the high-end, running on a 24-way iSeries model, Domino for iSeries provides the processing strength of 27 partitions, accommodating 100,500 active mail users in a NotesBench audit¹. That is 27 Domino for iSeries servers on one IBM iSeries! Each separate partitioned server can provide one or more functions of Domino for iSeries.

Many customers run multiple Domino servers on a single iSeries. A typical example involves multiple Domino servers where one Domino server or partition provides mail function, another application serving, and another HTTP serving. Scalability is extremely important since the workload on any Domino server increases. Disk space demand increases rapidly as e-mail users become more proficient with managing their messages and calendars and Domino applications are added to the system. As the sizes of Domino databases increase, the server view indexer tasks need more disk space and CPU cycles to run. During the life cycle of the iSeries server, you may need to add memory, disk capacity, and additional processors. There is no better platform to address these growth issues than the iSeries server.

2.1.2 Reliability and availability

Lotus Domino for iSeries takes advantage of the reliability and availability features of iSeries, such as RAID-5, mirrored disk units, and integrated backup capability. Each Domino for iSeries server runs as an application in its own subsystem. The unique iSeries architecture makes it safe to run Domino for iSeries and mission-critical business applications on the same machine. With a 99.97% proven availability rating, the iSeries is the most available server in the world for running Domino applications! On top of this, there are additional iSeries server hardware features to help keep the server running. An uninterruptible power supply (UPS) and backup procedures are also recommended to ensure maximum reliability.

2.1.3 Integration

Domino for iSeries includes integration between Domino and DB2 UDB for iSeries databases. Domino for iSeries maximizes its integrated access to OS/400, allowing direct, real-time access to DB2 UDB for iSeries. Data can be accessed through several methods and provide tight integration between Domino and DB2 UDB for iSeries.

Domino has a very rich set of integration options available for you to choose from. You can use one of the graphical tools that allow you to map your Domino environment to DB2 or program your own application logic. The tools include Domino Enterprise Connection Services (DECS) and Lotus Enterprise Integrator (LEI). If you choose to use one of the programming options, you can choose from the @DB functions, LotusScript:Data Object (LS:DO), Lotus Connector LotusScript Extensions (LC LSX), or Java Database Connectivity (JDBC). For more details on the performance aspects of these various integration options, refer to Chapter 9, "Integration with DB2 performance tips and techniques" on page 259.

¹ See <http://www.notesbench.org>

2.1.4 Proven security

Integrated, flexible security is a strength of both Domino and the iSeries server. Domino has built-in security features such as execution control lists (ECL) and access control lists (ACL). In addition, Domino security policies can be setup on the server along with access controlled directly in the Domino server document.

iSeries comes with support for some of the functions that would traditionally be found in a fire wall: IP Packet filtering, Network Address Translation (NAT), and SOCKS support. The iSeries server provides the flexibility you need in today's networked world to have function and security right where you need it.

2.2 Planning a Domino for iSeries server

Are you ready to move to the latest Web server technology known as Domino? Are you ready to jump right in and install the software? Well, not just yet. There are still some *significant steps* in *planning* that you need to accomplish before you implement your Domino for iSeries environment. There are several key areas that you want to be sure to include. Due to the scalability of the iSeries server, along with rich function sets available in Domino for iSeries, planning enables increased performance and application flexibility. The functionality of Domino for iSeries requires sufficient planning and testing.

While the product is easy to install, the science is in determining the organizational requirements and architecting Domino for iSeries to best meet those organizational requirements. As you find more uses for Domino, you need to account for the resources it takes to support those new uses. This is why planning and preparation are so important. As you will notice in the following chapters, the functionality of Domino for iSeries is broad.

2.2.1 Developing a Notes naming structure

Hierarchical naming schemes are the backbone to designing a solid Domino for iSeries infrastructure. One of the first planning steps to address is designing your naming structure. Do you want to have several organizational units or just one? The impact of this decision alone affects how you implement Domino for iSeries.

The hierarchical naming convention used by Domino is common name (CN)/organizational unit (OU)/organization (O). An example of this naming convention is Jane Doe/Southeast/Widgets, where *Jane Doe* is the common name, *Southeast* is the Organizational Unit, and *Widgets* is the organization name (Figure 2-1).

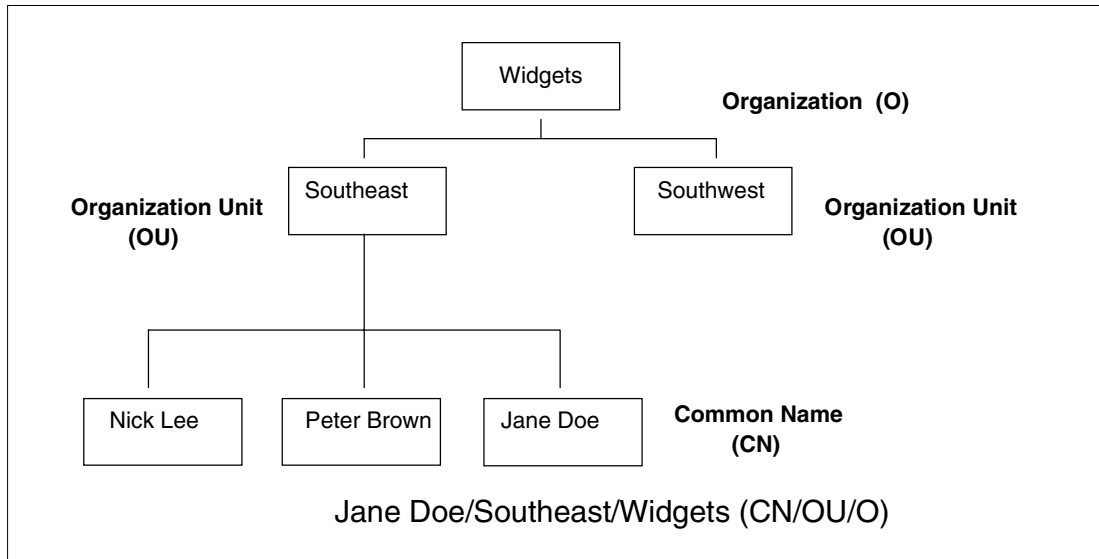


Figure 2-1 Canonical format for names

Only one common name and one organization are allowed. However, you can have multiple organizational units in your naming scheme. The OU is optional. Therefore, a naming scheme of CN/O is perfectly acceptable for small companies with a single Domino for iSeries administrator. However, companies are in the business to grow. Planning for company acquisitions and growth needs to be in the equation. When planning for common names, generally shorter names provide for greater ease of use.

Before you implement your hierarchical naming scheme, it is important to do some planning. Draw a pyramid with the organization name at the top, and one or more levels of OUs below, as shown in Figure 2-1. Each level in the pyramid represents one OU certifier ID. Maintaining many certifiers and passwords can be a significant administrative task on the CPU. Multiple OUs also create excessively long Domino names that contain multiple slashes.

Using departments as OUs has both advantages and disadvantages. It can be difficult to create and maintain an OU for each department because people can move between departments frequently. However, since employees in the same department work together and use the same databases and send e-mail to one another, the advantage of departments as OUs is their use as implicit groups for security and possibly for mail.

Alternatively, you may consider a hierarchy that consists of a company, division, and geographic location, or perhaps a naming scheme that aligns with your Notes system administrators. For example, if your Domino administrators are distributed geographically, you may want to create an OU for each location.

The naming convention provides a logical security grouping and a potential distribution of the administrative functions within an enterprise.

2.2.2 Choosing the right replication topology

Replication and mail routing are highly used functions on the Domino for iSeries server. Designing a replication and mail routing topology is a major step in planning your Domino for iSeries environment. Engineering an efficient topology allows your Domino for iSeries server to perform at its highest level of reliability and efficiency while minimizing costs.

The factors you need to consider when choosing a replication topology are:

- ▶ **Size of your Notes and Domino Network:** How many servers will be required to accommodate the number of users?
- ▶ **Location of your Notes and Domino servers:** Are all your servers scattered in several locations or are most of them in one building? Perhaps in one iSeries server?
- ▶ **Amount and size of the e-mail data being sent:** Is the average database (Notes Storage File) size only 1 or 2 MB in size or over 50 to 100 MB in size?
- ▶ **Keeping data current:** Is there a need to keep your databases current or can databases go without replication for a period of days?
- ▶ **The number of clients that will access the data:** Does your organization have several hundred or several thousand users per server?

If you plan your Domino topology early, your organization's environment will run more efficiently. You will also be prepared for unlimited future growth. Such issues as mail routing and replication are primary methods used for passing information between Domino servers. They are scheduled tasks that can consume many CPU cycles.

2.2.3 How security relates

Lotus Domino includes many layers of security. These layers include keeping your data secure on the Internet, mail routing, and replication within your organization.

The categories of security are:

- ▶ Server authentication
- ▶ Database access control lists (ACL)
- ▶ Form/view access control lists
- ▶ Document level access
- ▶ Field level encryption

2.2.4 What remains

The following list contains more questions that you need to answer:

- ▶ Will you be accessing the Internet? If so, you should consider using a firewall.
- ▶ Do you need encryption capability? If so, secure sockets layer (SSL) needs to be enabled.
- ▶ Where will your Domino for iSeries server be located? Is it behind a firewall?
- ▶ How do you register new users, and what are the key databases you need to be aware of when planning registration?
- ▶ Should you use advanced Domino services such as clustering, partitioning, or billing?

We attempt to address these questions throughout this redbook. Your decision to implement decentralized or centralized administrative functions is fundamental to this discussion.

2.3 The value of sizing, performance analysis, and tuning

Figure 2-2 represents the framework for this redbook. In addition, it provides a methodology for upfront sizing, long-term performance analysis, and tuning. We explore the methodology in greater detail in Chapter 3, “Sizing Domino for iSeries using the Workload Estimator” on page 19.

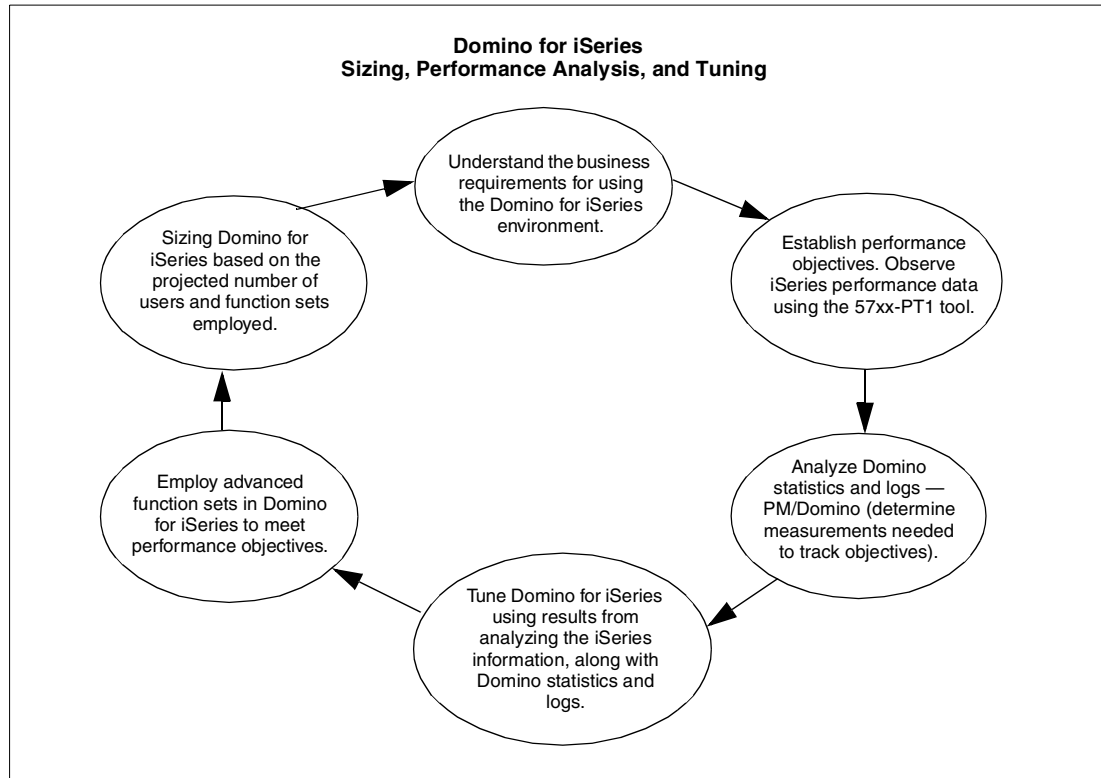


Figure 2-2 Method — Performance, tuning, and capacity planning

This book provides information about the implementation, management, and control of Domino for iSeries. You may ask why we are including implementation in this book. The functionality of Lotus Domino is quite diverse when compared to other applications. As a result, how the Domino for iSeries environment is architected or implemented is imperative. Never lose sight of the goal of performance monitoring in an organization, which is:

To develop an estimate of the system resources required to deliver performance in accordance with a documented Service Level Agreement and based on a forecast level of business activity at some specified date in the future.

Your ability to performance tune and size Domino for iSeries is directly related to how it is implemented and how it has been architected.

As you may imagine, Domino for iSeries has many features and functions. Each of these directly impacts the performance of the server in different ways. Due to the scalability and reliability of the iSeries server, many customers deploy all or many of the function sets of Domino on a single iSeries server. Knowing the impact that this type of architecture has on the system is necessary to understand what the system is or is not doing at any one time. Implementing Domino for iSeries on a single iSeries server is different from implementing Domino on several distributed Windows/NT servers.

Because of the iSeries subsystem architecture, multiple Domino servers can be very easily deployed on a single iSeries server.

The iSeries server is the most popular multi-user business computing system in the world. Lotus Domino is the most popular groupware solution. The three pillars forming the foundation upon which Domino for iSeries is built supports the pervasive theme of “simplicity through integration”. These pillars are:

- ▶ **Reliability:** 99.97% uptime, use of subsystems, automatic recovery
- ▶ **Scalability:** The most scalable Domino servers available today — up, down, and across
- ▶ **Integration:** Database, administration, mail, directory synchronization, save, and restore

2.3.1 Audience

The intent of this book is to provide traditional iSeries server administrators a methodology to manage the performance and capacity of their Domino for iSeries environment. It attempts to outline the issues involved in implementing the many functions of Domino while providing maximum throughput. In addition, it provides a framework for upfront sizing, long-term performance analysis, and tuning.

Traditionally, the iSeries server administrator evaluates resources in two primary environments of the iSeries server: interactive and batch workloads. Through the management of system resources in each of these environments, the server administrator provides better throughput for the end users. The same approach applies to Domino end users. As Domino functions are deployed, their impact on system resources can be measured and analyzed. Consequently, the architecture or implementation of Domino functions on the iSeries server is critical in understanding how the system is performing or will perform in the future.

Due to the functionality of Domino for iSeries, this book is primarily directed toward the traditional iSeries server administrator. However, the content of the book provides a much needed understanding to Domino administrators on how a particular set of functions can be better deployed on the iSeries server.

2.4 Domino architecture

Figure 2-3 illustrates the layered architecture of Lotus Domino and the large amount of functionality it delivers. Lotus Domino offers much more flexibility than simply an e-mail application.

The layered view accentuates the tight integration of core services and applications. It contrasts sharply with other products that are assembled from disparate sources and patched together with ad hoc interfaces.

This section and the following sections outline a few specific examples of services at different layers of the architecture to show the design.

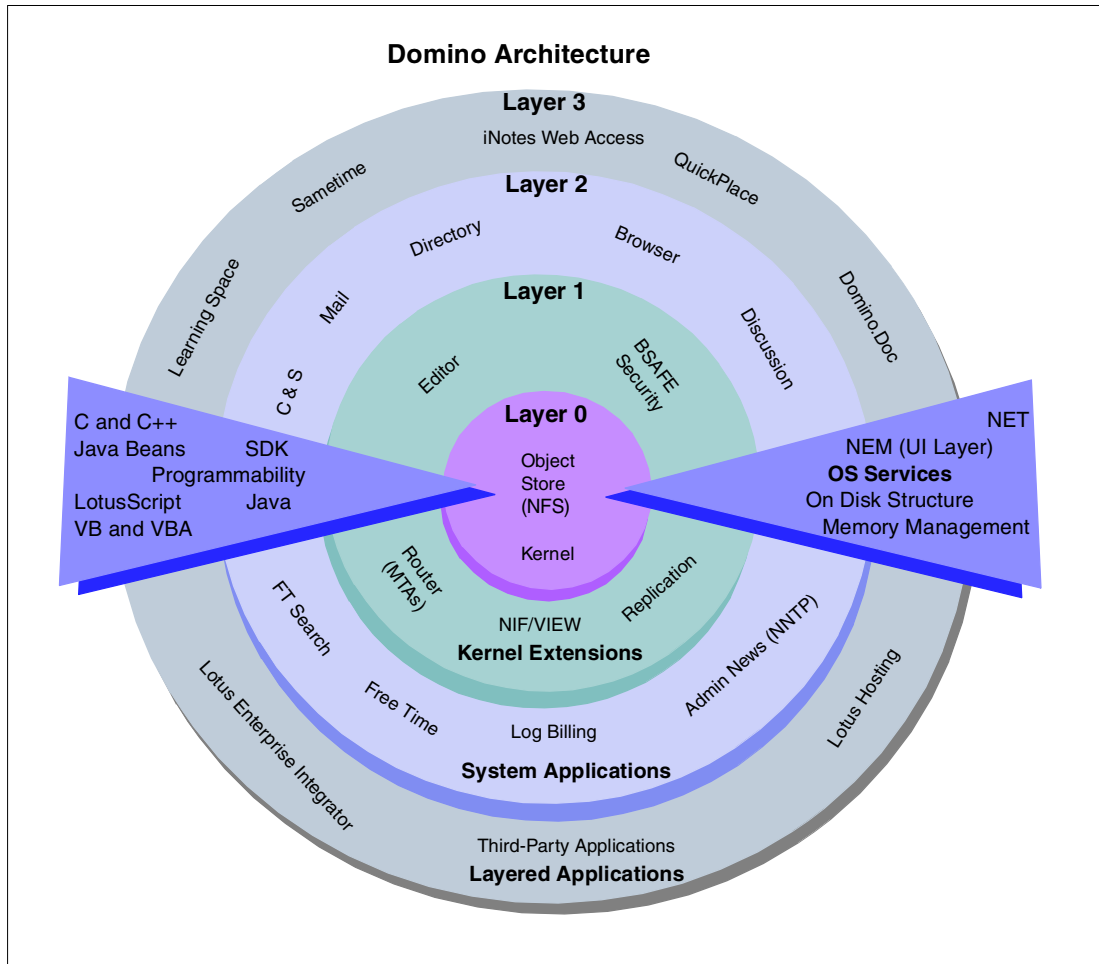


Figure 2-3 Understanding the Domino for iSeries architecture

2.4.1 Layer 0: The kernel

Layer 0 contains the document-centric object store. Each document, or *note*², is a compound structure of mixed data types arranged in fields that can be arbitrarily modified and extended. A document may contain text, rich text, binary blobs (attachments, ActiveX or Java Applets, for instance), encryption keys, doclinks, and so on. Each Domino database contains a collection of documents, and includes meta-organizing structures for display, security, retrieval, and access rights to the individual documents.

2.4.2 Layer 1: Kernel extensions

Layer 1 is logically distinct from Layer 0. However, it is tightly bound to the object store. Critical platform services are encapsulated within this layer. These include security, replication, routing, formula languages, view indexing, and document editing and rendering.

² A single document as seen by a Notes user is normally stored as multiple notes.

Each Layer 1 module is self-contained, with common interfaces to each other and the object store. As a result, new modules can be introduced into Layer 1 or existing ones swapped out. For example, the SMTP Message Transfer Agent (MTA), the router for Internet mail, was inserted in Release 4 to provide an alternative mail router. Later, with Domino R5, a native solution for SMTP was implemented to provide reliable and scalable mail router. The same ability can be used for search engines, security key rings (Public Key Infrastructures), and so on, as your needs require the adoption of new or standards-based components.

2.4.3 Layer 2: System and core applications

Layer 2 includes core collaborative applications that are coordinated by supporting system applications. Core applications (mail, calendar and scheduling, discussion, document management, and so on) are the first level of services directly visible to end users. As standard Domino applications, they can be modified and extended by Domino developers, but usually depend on kernel routines. System applications control and monitor these and all other applications. They include the LDAP-compliant directory service (Domino Directory), Logging, Administration, Billing, Free Time Lookup, and others.

2.4.4 Layer 3: Layered end-user applications

Layer 3 applications are not distinguishable technically from Layer 2 applications. Layer 3 applications also rely on the kernel and the *Domino Directory* (formerly called *Public Address Book* or *Names and Address Book*). However, most of them further extend core mail, calendar and scheduling, discussion features, workflow into higher level, task-specific applications.

The growing set of Domino applications shows the way complex functions evolve from lower architectural layers, while adding design and user interface elements of their own. Thousands of existing third-party Notes applications also fit into this layer. An example is a Sales Force Automation Domino application that utilizes security, workflow, document management, and SameTime to generate self-help Web applications for end users.

2.5 Tools available

There are a number of tools for sizing, performance analysis, and tuning that are available for the iSeries Domino environment. This section lists a few of the tools that are available:

- ▶ **iSeries and Domino tools**
 - Collection Services
 - Performance Management (PM/400)
 - Performance Tools/400
 - Performance Explorer
 - Management Central
 - BEST/1
 - Server.Load
- ▶ **Third-party tools for iSeries and Domino**
 - PM/Domino
 - Performance Navigator
 - NotesBench
 - GroupSizr
 - WEBSizr
 - LoadRunner
 - Tivoli

Comparisons between the native iSeries statistics and the Lotus Domino statistics are further explained throughout this redbook.

2.5.1 Collection Services

Collection Services is a data collection tool that is native to the iSeries. It can provide significant help in troubleshooting performance problems. Data analysis can be performed against this data collection online or offline. More information is provided on this important function of OS/400 in Chapter 4, “Basic concepts of performance analysis” on page 53.

Collection Services, formerly known as Performance Monitor, is a part of the operating system OS/400. To analyze the data acquired by collection services, utilize Management Central (see 2.5.5, “Management Central” on page 13, and 4.12, “Management Central performance monitor” on page 89) or the Performance Tools/400, Licensed Program Product (LPP) 5722-PT1 (or 5769-PT1 before V5R1). See 2.5.3, “Performance Tools (5722-PT1)” on page 12, and 4.13, “Reviewing data using Performance Tools/400” on page 114, for more information.

2.5.2 Performance Management/400

Performance Management/400 (PM/400) presents a broad array of benefits to you. As a systems management tool, PM/400 ensures that you get the most from your system by regularly monitoring performance and growth. These measurements allow you, your IBM Business Partner, or IBM to more easily plan for future system growth and identify potential resource constraints. Measuring system performance helps you make decisions that affect your budget and human resource plans.

PM/400 is automated and self-managing, which allows for ease of use. PM/400 gathers the non-proprietary performance and capacity data from your iSeries server and sends it to IBM. At a cost, you can receive this data in a series of reports and graphs that show the growth and performance of your system. You can view less detailed data for free online at:

<http://www-1.ibm.com/servers/eserver/iseries/pm400/>

PM/400 is automatically installed on your iSeries server or AS/400 system. It is more than a service; it is a partnership between you and IBM or your IBM Business Partner. This partnership helps you achieve the best return from your iSeries investment. Contact your service provider for more information. You can also visit the Web site at:

<http://www-1.ibm.com/services/its/us/mus14d1b.html>

2.5.3 Performance Tools (5722-PT1)

Performance Tools/400 (5722-PT1) manages and analyzes performance data, providing real-time and long-term views, as well as detailed analyses. You can order this product with one of two features: the Manager feature or the Agent feature. The Agent feature contains a subset of the functions in the Manager feature. Performance Tools has the capability to report, analyze, and model system performance. The Advisor function can analyze the system and give recommendations to improve performance.

The performance capabilities in both the Manager and Agent features allow you to:

- ▶ Manage performance data
- ▶ Monitor performance in real-time interactively
- ▶ Display performance data
- ▶ Analyze performance data
- ▶ Create a BEST/1 capacity planning growth model

Performance Tools/400 is a licensed software product from IBM. The costs varies depending on the processor group.

2.5.4 iSeries Performance Explorer (PEX)

The iSeries Performance Explorer (PEX) is a set of performance collection and reporting functions and commands that became available on the RISC versions of OS/400. The PEX functions, which are part of a total iSeries performance management methodology, should not be the first set of performance tools used when analyzing a performance problem. The Performance Explorer and Collection Services are separate from one another and produce two completely different sets of database files that contain grouped sets of collected data. Both collections of data can run at the same time. However, this should be kept to a minimum because the impact to the system is significant with both collections active. As with Collection Services, the capability to collect Performance Explorer data is shipped with OS/400. The ability to analyze this data requires the purchase of Performance Tools for AS/400, LPP 5722-PT1.

You can find further information on general PEX usage in 4.14, "Performance Explorer (PEX)" on page 116. You can also find more information for specific PEX trace points for Domino in 4.14.5, "Performance Explorer trace points for Domino" on page 117.

2.5.5 Management Central

Management Central monitors provide the ability to monitor system-wide metrics, as well as focus in on specific jobs. Prior to V5R1, the monitors were classified in one group that monitored various performance statistics across the system. These monitors are now called *system monitors*. System monitors provide real-time monitoring along with the ability to view historical data on such areas as disk arm utilization and interactive feature utilization to name a few. The job monitors can monitor the activity of a single job or numerous jobs by job name, user, job name, subsystem or server type. You can learn more about Management Central monitors in 4.12, "Management Central performance monitor" on page 89.

2.5.6 BEST/1

The BEST/1 Capacity Planning tool can be used today to create models of various workloads to estimate the processor requirements along with disk, memory, input/output processors (IOP), and other requirements. This tool contains two predefined workloads for Domino that are based on NotesBench. It is best to collect your own performance data to feed into the BEST/1 tool for accurate capacity planning results. BEST/1 is part of the OS/400 Performance Tools Licensed Program Product (5722-PT1).

As with every performance estimate (whether a rule of thumb or a sophisticated model), you always need to remember that it is an *estimate*. This is especially true with a robust product, like Domino for iSeries, that offers so many different capabilities with a chameleon-like ability to conform to characteristics and demands. The typical disclaimers that go with any performance estimate, such as "your experience may vary", are especially true with Domino.

Note: After OS/400 V5R1, BEST/1 will no longer be a part of Performance Tools and will no longer be an offered LPP for OS/400.

2.5.7 Server.Load

Lotus Server.Load is a tool that measures and simulates various Domino server capacity and response metrics. Server.Load can be used to simulate the load that Notes client-based applications place on a Domino server. It allows you to select a workload and run it against a target server. The workloads (also called *tests* or *scripts* in this redbook) simulate the behavior of a Domino workstation to server.

Server.Load offers Domino administrators the flexibility of selecting a built-in script, creating a custom script from a library of commands, or submitting commands manually. For example, you can select to run the built-in Simple Mail Routing script and simulate users on a client reading and sending mail. You can also create a custom script that creates and opens a Notes mail database and populates it with notes. If you want to test or run individual commands, such as deleting notes from a database or even running remote server console commands, you can do this as well.

Server.Load also enables you to monitor server metrics in real time by using the Real Time monitor window. Server.Load provides the following features:

- ▶ An easy to use interface for running and customizing scripts and measuring server performance
- ▶ The choice of using built-in or custom scripts for the workloads
- ▶ A command line to submit manual commands
- ▶ Real-time control of test environment and variables
- ▶ Real-time characterization of server performance and capacity metrics
- ▶ Server.Load help database

Starting with R5.06, Server.Load is shipped as part of the client CD or download kit. It consists of the tool `sload.exe` and the documentation `sload50.nsf`, which can be located in the Notes data directory when installed. Server.Load installs as part of the Administration Client.

2.5.8 Performance Navigator

Performance Navigator is a graphical PC Tool for iSeries Performance Management and Capacity Planning that runs on Windows 95, 98, NT, and 2000. It has an intuitive interface combined with advanced statistical capabilities to access the iSeries performance data and reports dozens of performance metrics. It has a dynamic linear regression algorithm for trend analysis, capacity planning, and growth prediction. It enables you to sort through the data by month, week, day, and interval. This is a chargeable program product that runs on your workstation to analyze the QMPGDATA library from the iSeries server.

For more information, see the Web site: <http://www.mpginc.com>

2.5.9 NotesBench

NotesBench provides a collection of both benchmarks and workloads for evaluating the performance of Notes R5 servers. NotesBench is a third-party consortium organization with a mission to measure vendor server platform performance with Lotus Domino servers. NotesBench is available to hardware vendors only. The statistics can be used for such applications as Server.Planner. The workloads are software components that simulate the behavior of Notes workstation-to-server or server-to-server operations. They return measurements that let you evaluate server performance in relation to the cost of ownership of the server.

NotesBench provides:

- ▶ A command-line user interface for running the benchmarks
- ▶ Scripts used in the benchmarks
- ▶ Special variables for notes.ini files
- ▶ A checklist of disclosure information
- ▶ A checklist for auditors
- ▶ The *Using NotesBench* guide
- ▶ Related Domino and Notes documentation

NotesBench gives you an objective way to evaluate the performance of different platforms running Notes. It is intended for testing engineers at hardware and software vendors. NotesBench requires that vendors run the NotesBench tests and publish their performance results. Each vendor runs the same tests in the same manner, and the tests can be audited. For more information, see the Web site: <http://www.notesbench.org>

2.5.10 GroupSizr

GroupSizr (formerly known as and NoteSizr) from Technovations is an application used to size, benchmark, and analyze Lotus Notes servers, systems, and applications. GroupSizr can be used to simulate many real Lotus Notes clients using a single PC. The tool is capable of running a sequence of commands against any specified server or set of servers and any given database and any Notes application.

GroupSizr produces concise and easy-to-understand results and reports, in addition to providing run-time analysis of test performance data. GroupSizr is particularly suitable for simulating a large amount of session requests within a limited duration of time, for example, on Monday at 8:00 a.m. This is a PC-based Licensed Program Product. However, you can download an evaluation copy by visiting the Web site at: <http://www.technovations.com>

2.5.11 WEBSizr

WebSizr is a performance analysis and sizing framework for HTTP-based servers and applications. Application developers use WebSizr to engineer high-performance Web applications. WebSizr helps the architect and systems engineer to easily characterize the performance, scalability, and sizing of:

- ▶ HTTP-based Web servers
- ▶ Internet or intranet applications
- ▶ Electronic commerce applications

This is a PC-based licensed program product. However, you can download an evaluation copy by going to the Web site at: <http://www.technovations.com>

2.5.12 LoadRunner

LoadRunner is a load testing tool that predicts system behavior and performance. It exercises an entire enterprise infrastructure by emulating thousands of users to isolate problems, optimize performance, and accelerate deployment.

The Virtual User Generator lets you record both two-tier and three-tier client/server traffic into a clear and readable script. The dynamic monitoring of all virtual users allows you to organize and control how they test individually and how they interact during a scenario. The server monitor allows you to observe the database or application server during the load test. Summary statistics are presented with clear, attractive graphs and reports that help analyze load scenario results and pinpoint performance bottlenecks.

LoadRunner is a licensed product. While the costs for this product is rather expensive, it provides a wide array of functionality. For more information, visit the site at:
<http://www-heva.mercuryinteractive.com/products/loadrunner/>

2.5.13 Tivoli Manager for Domino

Tivoli Manager for Domino provides the ability to monitor hundreds of Domino servers from a real-time graphical interface, all of which can be done from a single console. The latest release of Tivoli Manager for Domino is version 3.1. The latest release contains many new enhancements that include:

- ▶ Web-based interface
- ▶ Additional monitors
- ▶ Added features for performance stations
- ▶ Integration with Tivoli Data Protection
- ▶ New graphic and reporting tools

You can find more information and news updates on Tivoli Manager for Domino at:

- ▶ <http://www.tivoli.com>
- ▶ <http://www.tivoli.com/news>

2.5.14 Tivoli Application Response Measurement (ARM)

The Application Response Measurement API is a set of standard API calls, agreed upon by a number of prominent IT companies, that allow an application-centric focus on performance.

The ARM API allows you to monitor the network that connects a client to a server and record the interval from when a request leaves the client until a response arrives at the client. You can use a program that generates a typical client workload and make time measurements within the program. You can also measure the utilization for all the hardware components involved in providing service to the user. And you can assume that if these values are at reasonable levels, the user response time must be acceptable.

This tool is a part of the "Enterprise" of Tivoli tools. For more information, visit:
http://www.tivoli.com/support/downloads/arm_sdk/armguide.html

ARM is intended for the application developers who wants to know how to instrument an application for transaction monitoring using the standard Application Response Measurement API function calls.

Version 2.0 of the ARM Software Developer's Kit has been developed with the help of the ARM Working Group of the Computer Measurement Group (CMG). The 2.0 ARM SDK, including this documentation, is available on CD and from the CMG Web site:
<http://www.cmg.org/regions/cmgarml>

This Web site also contains information on performance measurement agents that use the data generated by the ARM API function calls and the latest information regarding future updates or changes to the API.

A public discussion list for ARM, *cmgarm* is now available by sending e-mail to:
cmgarm@cmg.org

To subscribe, send the following message to majordomo@cmg.org:

subscribe cmgarm

2.6 NSD for OS/400 (Dump utility)

The Notes Server Dump (NSD) is a Lotus tool designed to gather information about a failing Domino Server. The tool extrapolates the critical data during the life of the Domino server. Upon failure, the tool produces a *dump* file. This file, when presented to Lotus support is intended to provide all the Domino information a Lotus Quality Engineer needs to troubleshoot just why the server failed. This was originally designed to be run with various UNIX systems. The NSD tool has been ported to run on OS/400 to diagnose Domino for iSeries. The tool collects and provides the following data to Lotus:

- ▶ Active jobs
- ▶ Environment variables
- ▶ Job call stack
- ▶ Job locks
- ▶ Display open files
- ▶ Dump of semaphores, shared memory, and message queues
- ▶ Dump of notes.ini file
- ▶ Commitment control status
- ▶ Display communications status
- ▶ Job mutexes (can be seen as *locks*)
- ▶ Job activation group

2.6.1 Activating the NSD tool

NSD can be activated by editing the notes.ini file and adding the line:

```
CleanupScriptPath=call qnotes/nsd
```

When the server is started, the information is dumped to a Stream File (STMF) in the Data directory of the Domino server. The file name is *nsd_all_os400_(julian date)_(time)*. Consider this example *nsd_all_os400_19980115_15:22:21*.

The file name is often truncated on a green-screen display because it is relatively long. In a Client Access green screen, if you click the file and then click the F22 key, you see the full file name.



Sizing Domino for iSeries using the Workload Estimator

This chapter is designed to offer guidelines to help use all of the functionality of the IBM Workload Estimator for iSeries (WLE) when sizing Domino to arrive at the most accurate estimate possible. Please keep in mind that even after reading this chapter and having a better understanding of how to use the WLE to achieve a more accurate estimate, it is necessary that you *treat the results and recommendations* of the Workload Estimator as *estimates*.

For more detailed sizing information, we encourage you to use the BEST/1 modeling tool. The BEST/1 tool is designed for performance specialists (including customers) to be used for complete capacity planning. The BEST/1 capacity planner is part of the IBM Performance Tools/400 product, which is a licensed program for the iSeries and AS/400. For more information, see Appendix A, “Capacity planning for Domino server using BEST/1” on page 391.

Another alternative is to use the Performance Management/400 integrated with the Workload Estimator (see 2.5.2, “Performance Management/400” on page 12, for more information).

3.1 Introduction to Workload Estimator

The purpose of the IBM Workload Estimator for iSeries (WLE) is to provide recommendations for estimating an appropriate configuration for an iSeries or an AS/400e server running one or more workloads associated with e-business or collaboration, such as Domino. It is a Java servlet-based Web application on the Internet that you can find at:

<http://www.ibm.com/eserver/iseries/support/estimator/index.html>¹

Tip: Before you start using the Workload Estimator for the first time, we highly recommend that you print and read carefully the help documentation, which opens up automatically in a separate browser window after you accept the page with conditions of usage (see Figure 3-1) as well as through the tutorial provided with the tool. The tutorials can be downloaded as PDF files.

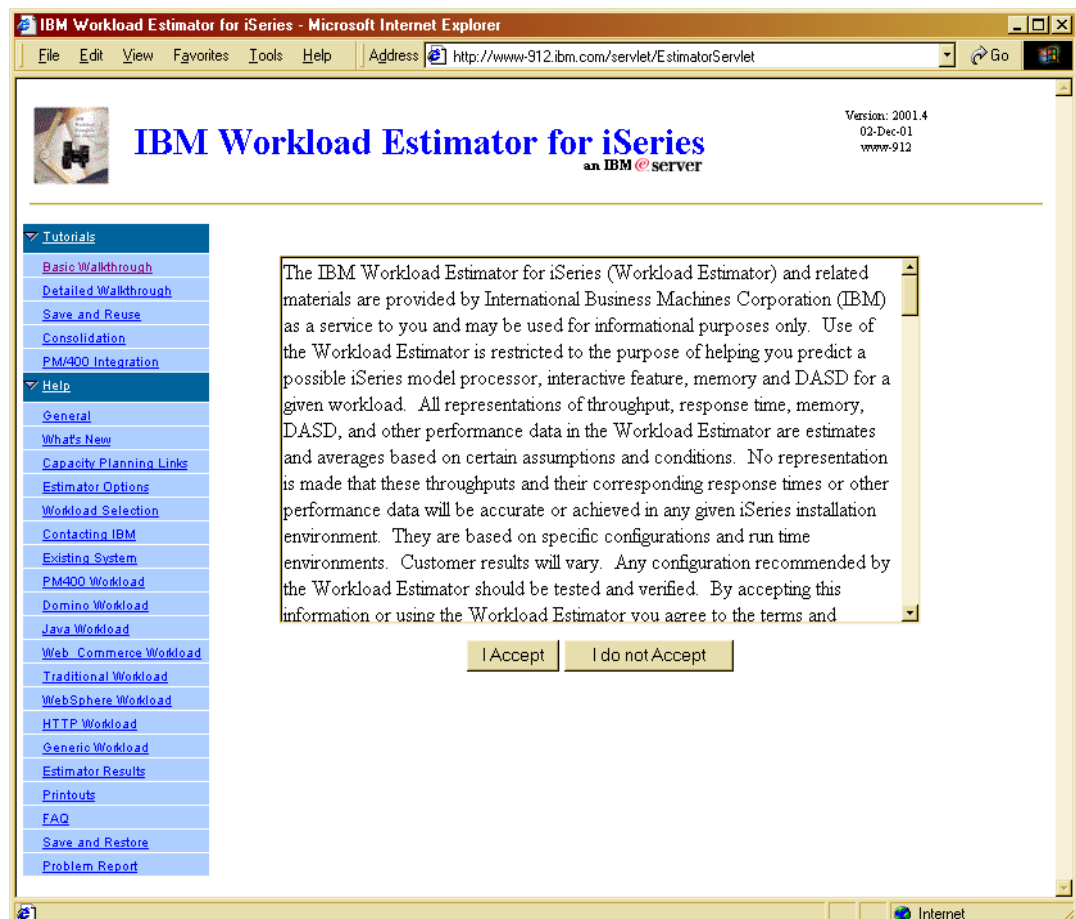


Figure 3-1 Workload Estimator conditions of usage

The majority of the use of the Workload Estimator is to size new iSeries or AS/400e models. It is, however, possible to estimate for an upgrade of an existing system with a pre-existing system load by adding additional work.

There are two ways to enter data about an existing system into the Workload Estimator:

¹ This URL has been directed recently to be consistent with other iSeries Web pages. The former URL <http://as400service.ibm.com/estimator> will redirect you to the current site.

- ▶ You can select the Existing workload option on the Workload Selection page and complete the questions asked about the existing resources of the system.
- ▶ Import existing system data directly into the Workload Estimator from Performance Management/400 (PM/400).

The recommendation or *Selected System* reported by the Workload Estimator includes the model, processor, interactive feature, CPW capacity, memory, and then number and capacity of disk storage (DASD) necessary to support the set of workloads defined by the user. When providing the sizing recommendation, the Estimator always defaults to recommend the latest technology systems that are available.

Even though these detailed resource requirements for each WLE recommendation are given, it is important to remember that just as with every performance estimate (whether a rule of thumb or a sophisticated model), you always need to treat it as an estimate and not a detailed capacity planner. This is particularly true with a robust system like the iSeries server that offers so many different capabilities where each installation will have unique performance characteristics and demands.

The sizing recommendations that are generated by the Workload Estimator began with benchmark measurements and performance tests based on well-defined repeatable workloads. For Domino, most of the measurements that were performed were the NotesBench consortium benchmark tests. Through the experience of running these measurements with large numbers of users, as well as feedback from internal and external customers and business partners, certain performance projections for each resource have been developed. These are used in the calculation for a recommended system.

To arrive at a recommended system, the Workload Estimator goes through a series of questions to collect the necessary information to calculate the resource requirements of your solution. As you become more familiar with the Workload Estimator, you can provide more specific information for particular workloads, thereby leading to a more accurate estimation.

The Workload Estimator is constantly being updated to implement the latest versions and releases of the workloads. New function is being added on a regular basis to keep the information and projections current. Periodically review the online version of the Workload Estimator to keep up-to-date with the most accurate information at <http://www.ibm.com/eserver/iseries/support/supporthome.nsf/document/16533356>, which always describes the latest changes.

3.2 Benchmarks to determine the load of specific workloads

A few of the factors needed to determine a system's overall impact are the number of active or concurrent users, the type of work being performed, the frequency by which transactions are performed by the users, and the complexity of the applications being deployed.

Although there are other factors to consider to determine the total impact on the system (communication lines, I/O processors, bus speeds), the parameters mentioned in the previous paragraph are the critical factors for the Workload Estimator. A series of questions are asked in WLE to acquire a combination of these factors to determine the resource requirements to support the load of your solution.

Because of the diversity of business applications being deployed today, the best way to appropriately size a system for a given application is to test the behavior of many system configurations to see how they perform. Testing would require extensive hardware and people resources. Because of this, detailed testing is not typically performed by customers or IBM business partners.

A practical alternative is to select a workload that defines the type and frequency of the transactions performed by users and use this workload to put a load on a system. After the workload is defined, you can then extrapolate the results to a real user environment without having to run the test on many different system configurations to see how they each perform.

This is only meaningful in conjunction with the well-defined workload for the benchmark. In most cases, the benchmark result does not correlate to the exact type of “real” users working in your organization. An example of this is the *NotesBench workload* used to measure the impact of mail users on a Domino server.

The NotesBench consortium was established to define standard workloads for measuring Lotus Domino and Notes Benchmarks on different platforms. One specific benchmark that is of interest when using the Workload Estimator, especially for sizing a messaging workload, is the *R5Mail workload*². The R5Mail script is as follows:

- ▶ Events occurring every 15 minutes
 - Read five documents
 - Update two documents
 - Delete two documents
 - Scroll a view once
 - Open and close one database
 - Open and close one view
 - Send one memo to three recipients
 - Do three lookups on the Domino directory
- ▶ Events occurring every 90 minutes
 - Schedule one appointment
 - Send one invitation

For many benchmarks being published today, there is a multiplication factor used to project from the benchmark “virtual” user to a “real” user. As a general rule of thumb, a NotesBench R5Mail user equals approximately one-half of the complexity of a typical mail user. This means that in the Estimator, the number of *typical users* supported by a system is equal to one half of the number of NotesBench users reported for the same system.

For example, if an iSeries Model 840-2461 was tested with 100,500 concurrent NotesBench users, then that same system should be able to support 50,250 typical mail users.

Note: Although the typical user comes much closer to an average user in reality than the NotesBench R5Mail user does, the workload generated by real users may deviate considerably! This is most significantly true when it comes to applications other than mail, but even the real life mail workload can differ substantially.

You can find the iSeries audited NotesBench results and other useful NotesBench information at: <http://www.notesbench.org>

Aside from viewing the existing iSeries and AS/400 benchmarks that have been published to help give you an idea of the data behind the Domino workload in the Estimator, this is also a good reference point to start when investigating possible migrations from other platforms. It is possible to compare the NotesBench results published from another platform for a specific workload with the results of the same workload on the iSeries to understand how to input your existing workload on another platform into the Estimator.

² Earlier NotesBench audits as well as sizing aids for Domino for AS/400 used a different workload based on Domino Release 4.5.

3.3 Workload Estimator terms

When using Workload Estimator for sizing a Domino solution, it is important to use WLE terms as they were intended for use within the tool. This section includes a discussion of a some terms that could be interpreted differently within the context of Domino.

3.3.1 Workload

On the Workload Selection page (shown in Figure 3-2), the term *workload* refers to the entire set of parameters defined within the pages that correspond to one of the types of workload selected. It is possible to select more than one of the same type of workload. This allows the user to size more than one Domino application per session of the Workload Estimator. Specifically to Domino, if mail and applications are selected, the workload, as the Estimator understands, is both of those Domino uses combined.

Because many different applications can make up your total Domino solution, the workload selection could include as many different Domino workloads as your total business solution needs to define all of the applications. Similarly, for a Domino mail solution, it is possible to select multiple Domino workloads to account for each of the types of mail access.

Alternatively, since each Domino workload that is defined in the Estimator contains both a mail profile and an application profile, it is possible to only have one Domino workload defined. For example, you could either define one Domino workload that details your messaging or mail solution and another Domino workload that details your end-to-end business application like a Web shopper. Or you could combine both into a single workload.

IBM Workload Estimator for iSeries - Microsoft Internet Explorer

File Edit View Favorites Tools Help Address http://www-912.ibm.com/servlet/EstimatorServlet Go

IBM Workload Estimator for iSeries
an IBM @ server

Version: 2001.4
01-Dec-01
www-912

File
Edit
Navigation
Contact IBM
Tutorials
Help

Workload Selection

Type of Workload	Name of Workload
Domino	Domino #1
Workload Type	Workload #2
Workload Type	Workload #3

Use the pull downs to select your workloads. Then press **Next**

Allow Another Workload

[Options: OS/400 Version = V5R1, RAID Support = None, DBCS Support = No](#)
[Developed by the Rochester iSeries System Performance Team](#)

Internet

Figure 3-2 Workload Selection page from Workload Estimator

3.3.2 Instance

Some Domino administrators use the term *instance* to refer to a Domino server. For example, you may have multiple instances of Domino on a single iSeries server. In Workload Estimator, the term instance in the phrase “the number of partitions being used by this instance of the Domino workload” (as shown in item 7 in Figure 3-3 on page 25) actually refers to specific Domino workloads defined in the Workload Estimator such as “Domino #1” in Figure 3-2.

This instance may be one of multiple applications deployed on the same Domino server or across multiple Domino servers. Therefore, in this reference to the word instance in the Workload Estimator, the user should think about the specific usage type that is being defined, not the instance of the Domino server.

3.4 Domino sizing concepts in the Workload Estimator

Due to the diverse Domino solutions being deployed, there are multiple ways to use Workload Estimator to size a Domino solution. The following sections describe in more detail certain Domino specific parameters that will help when you are faced with the task of estimating the load of a total Domino business solution and recommending an appropriate iSeries to support the solution.

The Domino workload within the Workload Estimator has questions designed to acquire the factors of your Domino environment to appropriately determine the recommended system. The default values are suggested values for brand new Domino customers who do not know how their users will end up using the application.

For those customers and business partners who are familiar with their Domino application and their user's habits, this section offers some definitions and helpful hints on how to interpret a few of the Domino workload concepts used to determine the recommended system.

Tip: We highly recommend that you look at every single parameter and try to understand its significance and how it might vary depending on your specific environment. The more often you accept the default values, the less likely the resulting suggested configuration will be useful.

3.4.1 Concurrent users

The number of *registered users* is the easiest to understand and always a known input variable. Note, however, even though this should be the first question to ask, there is *no way* to come to a proper estimation, if the number of users is the only information you have.

The number of *active* or *concurrent users* is very important when sizing the customer's solution. There are a couple of ways to report the total number of active users that a customer will be deploying.

The defaults for concurrency, as shown in Figure 3-3, are set to values that are suggested for new Domino customers who do not know the frequency that their users will use the applications. Again, if you accept the defaults, very likely it may not match your environment. If you know the percentage of users that are active during peak times, adjusting the concurrency rate for the application accordingly will result in a much more accurate recommendation.

Figure 3-3 Workload Estimator Mail Definition

When you adjust the concurrency ratings within WLE Domino workloads, it is important to understand how the concurrency ratings are used. By selecting one Domino workload, choosing both mail and applications, and leaving the Estimator defaults for concurrency, the highest total concurrency rating for all of your users is 65%. This is due to the 50% default mail concurrency and the 15% default application concurrency rating.

For example, if you have 1000 users and all users work with mail *and* a Domino application, then the mail and application profiles would each have 1000 registered users. WLE would then calculate using the concurrencies of each Domino usage type that 500 users would be using mail and 150 users would be using the Domino application concurrently. This totals 650 users actively interacting with the server at once. Sixty-five percent (650 out of 1000) of the users would then be active on the system driving the utilization.

It is probably not appropriate to specify each of the usage types as being 100% busy if you also specify that all 1000 users are registered users of mail, the user-written applications, and Domino.Doc. This would give a concurrency rating of 200%, which is very unlikely.

Table 3-1 shows the specific default values for concurrent users for each Domino usage type. The default concurrency values shown in Table 3-1 are based on feedback from IBM representatives and Business Partners who are familiar with the way that current customers are typically running as well as direct feedback from internal and external customers. Note, however, that these values may be significantly different to your own environment.

Table 3-1 Default concurrency values

Domino usage type	Default concurrency
Notes Client Mail access	50%
WebMail access	15%
IMAP Mail access	15%
POP3 Mail access	15%
iNotes Access for Outlook	15%
Applications	15%

In order to determine the total number of concurrent users for each Domino usage type being defined in Workload Estimator, use a 15 minute interval. This means that an active user is any user who has performed an activity that made a request to the server within a 15 minute interval. If a user was active at anytime in that 15 minute interval, they should be counted as a concurrent user. Please note that this value does not correspond to the default time-out (Server_Session_Timeout), where inactive users are automatically disconnected after the specified time.

If you use WLE for existing Domino installations, there are a couple of ways to help determine users' concurrency. For connected Notes Client users, SH STAT SERVER shows the peak number of users. This value would be a great starting point for the number of concurrent mail users. SH STAT SERVER does not show how much work the users are generating. Therefore, this would not be the place to determine users' complexity.

For Web type users, HTTP is connectionless from a Domino statistics standpoint and, therefore, the statistics cannot show any numbers for "connected users", but the value Domino.Threads.Active.Peak can give some indication of concurrency.

The 15 minute interval recommendation is a rule of thumb used to determine the number of concurrent users of the Domino application being deployed. This rule of thumb also applies to remote users. For example, if all of the remote users were replicating every 15 minutes, then they would be 100% concurrent. It is important to note in this circumstance that *disconnected users* should be classified as casual mail users.

3.4.2 Number of partitions

The number of *partitions* calculated in Workload Estimator refers to the number of *Domino partitions*. This metric does not refer to the number LPAR partitions being configured on the iSeries server.

By using the default of *CALC to determine the number of partitions, WLE uses a value of 3000 concurrently active users per Domino partition. Depending on the workload generated by each user, it may be possible to support more users per partition. The default of 3000 users per partition is a good starting point³.

³ With Domino R4.6 and before, there was a certain maximum workload which could be handled efficiently by each Domino server. This made it necessary for many administrator's to limit the number of mail users, per server. At the time, the recommendation was not to exceed 500 to 700 users per server. With Domino R5, the implementation has been changed, so each server can support a much higher workload.

There are of course instances in which overriding the Estimator default would be appropriate. In a case where you have a total of 3000 users, instead of putting all 3000 users into one partition, then you could put 1500 users in each of two partitions, to allow easier management of the servers. Especially for an application like mail, where each user owns a particular database, a server becomes difficult to manage if the number of databases exceeds a certain limit.

3.4.3 Clustering

It is important to note that clustering defined for a Domino workload is actually Domino *clustering*, not OS/400 clustering. Two or more Domino servers on the same or different hardware (and same or different operating systems) can be grouped as a cluster. Important for the workload is a task called *cluster replication* (see 8.3.3, “CLREPL: Cluster Replicator” on page 246). This task ensures that changes applied to a database that have a replica on a server in the same cluster are replicated immediately⁴.

When Domino databases are clustered, the server performs real-time replication for any changes that occur to the databases. Therefore, clustering introduces additional processing requirements that are used in sizing. For that reason, and since there are many ways to configure clustering, it is important to interpret and specify the appropriate information in WLE.

In reality, not all databases on clustered servers need to be enabled for cluster replication. The assumption for clustering in WLE, however, is whichever clustering configuration is chosen for a particular Domino workload instance applies to *all* of the databases defined in that instance. For example, if a Domino workload is defined with mail and applications, the clustering configuration is applied to the total number of all mail databases and all other databases on that server.

Your solution may be designed in such a way that clustering is deployed on a more granular level, so that mail and application databases are not clustered together. It is necessary to define multiple Domino workloads for each usage type and specify the appropriate clustering configuration for each. Similarly, if there are some user databases that are not clustered at all, you should define a separate Domino workload.

The “Clustering with other Domino Servers” option refers to both being clustered to another server and having another server clustered onto the system being sized. If the system is being used by another system for clustering, it is also necessary to fill in the amount of disk storage needed for clustering.

“Cluster on the same iSeries server” refers to the case where you have multiple Domino partitions on the same iSeries server using Domino clustering. If the partitions are clustered on the same system, the Estimator will automatically double the number of partitions that have been either calculated (*CALC) or that the user has specified.

In addition, the Estimator calculates the amount of disk storage needed for clustering by adding the amount of disk storage required for the total number of mail databases, as well as any application databases that are defined. Here is how WLE would calculate the resources needed for clustering based on the input to the specific clustering questions:

1 partition (3000 concurrent users)
+ 1 partition (clustered on the same system)
<hr/>
2 total partitions

⁴ As opposed to “normal” replication that only takes place periodically.

3000 mail databases (3000 concurrent users)	
+ 3000 mail databases (3000 replicated databases for clustering)	
<hr/>	
6000 total mail databases	

3.5 Domino mail concepts

As noted previously in the NotesBench benchmark discussion, it may not be completely accurate to extrapolate NotesBench mail users to real life users. The first thing to note is that WLE uses the NotesBench benchmark value for the number of users that will fit on a system. Then it divides that by 2.

Although this is done in an attempt to more accurately depict an average mail user, not all of the users working with mail server today are average or “typical”. Therefore, the user complexity question in Workload Estimator gives you the ability to further classify the mail users, as the casual, moderate, and heavy values are given a type of weighting factor.

The default values of 10 casual, 70 moderate, and 20 heavy mail users is a good starting point if the organization did not use Notes mail before. The default mixture was chosen to represent a “typical” mail environment. This “typical” user projected for a given system will be approximately equal to one half of the number of Mail and Calendaring users (MCU) specified for that system. For a more detailed discussion of the mail user metrics used for projections, see 3.10, “Additional tips and insights” on page 44.

If you use many agents and extra function with your mail solutions, there is only a fine line between a heavy mail user as defined in Workload Estimator and a Custom Application user. One possible solution to sizing a very complex messaging solution is to use the Application profile for Domino in WLE.

For example, a mail profile with 100% heavy Notes Client mail users represents the same load as an Application profile with an application rating of 1.3. If your mail user profile is very complex, with many views and agents being deployed, it may be appropriate to use the Domino Application profile.

Note: The mail profile should be used in most cases to estimate a mail workload. You should only consider using the Application profile in cases when you know your Domino mail solution very well and understand the differences to a standard mail environment. However, do not use the mail profile to estimate the load of a custom written or purchased Domino application.

Just as it is important to use caution when sizing very heavy mail customers, it is also important to avoid overestimating the number of casual users. The casual user does e-mail only, rarely sending or receiving attachments. Therefore, casual users can quickly become more complex, for example, by starting to use the calendaring and scheduling features as they become more experienced with Notes mail. New Notes users will generally advance from being casual users to moderate users in only a few months.

3.5.1 Mail access types

Another factor that has an impact on the workload generated by mail users is the access type. Mail access types currently supported in the Workload Estimator include:

- ▶ Lotus Notes Client
- ▶ iNotes Access for Microsoft Outlook

- ▶ iNotes Web Access
- ▶ WebMail
- ▶ IMAP
- ▶ POP3

As noted above, most of the mail access type profiles in Workload Estimator are based on information collected from internal benchmarks using the NotesBench suite of workloads. The recommendations made by WLE are not directly comparable with the audited NotesBench data.

Because the various NotesBench workloads create differing amounts of work per user, they have been normalized for the Workload Estimator. For a specific mail access type, the assumption is that each user would do the same amount of work within 15 minutes. The difference would be in the way the user's mail client connects to the server.

With the many choices of mail clients that are available, you could support multiple mail access types with your mail and calendaring solution. When using WLE to size such a heterogeneous solution, you use multiple Domino workloads to define each separate mail access type.

Important mail factors

The number of concurrent mail users has a significant affect on the overall recommendation for a Domino mail workload. Based on feedback from customers, the default concurrency ratings for each of the mail access types have been set to represent a typical mail environment.

For a new Domino mail user, it is usually appropriate to keep the defaults. If you are familiar with the way your mail users are working, see 3.4.1, “Concurrent users” on page 24, for general Domino sizing.

Extremely large databases

Another factor that has some affect on the performance and sizing of a Domino mail solution is the size of the actual mail databases being deployed along with the size of the Domino directory. It has been noted that as the size of these databases grows, the more complex the average user becomes, which results in heavier loads.

The disk storage per registered user needed for the mail databases parameter in WLE currently effects only the disk capacity recommended by the user. For extreme environments with gigantic mail databases (for example, over 500 MB) and a huge Domino directory (for example, over 1 GB), it is possible to account for the extra load by defining a *Generic Workload*. In the Generic Workload profile, there is an option to add processor resources as appropriate to any type of work being deployed.

The Generic Workload can be a very versatile workload. It provides a controlled manner in which the user is allowed to enter various raw parameters of a workload. These parameters include such things as the interactive and non-interactive CPW processing requirements of a workload, the amount of memory, the number of disk arms, and the amount of disk storage needed by the workload as shown in Figure 3-4.

Figure 3-4 Generic workload panel

When you define mail users with databases larger than 200 MB, it is most likely appropriate to use the heavy mail setting for those users.

Finally, as new versions of mail clients, Domino releases, and OS/400 releases are announced, measurements and analysis are done to keep WLE up-to-date with the latest technology being deployed on the iSeries. Therefore, it is important to continue to use the latest version of the WLE that is available.

3.6 Domino application concepts in Workload Estimator

When we talk about *Domino applications*, we mean custom written or purchased Domino databases that implement certain collaboration logic. The logic of those applications is typically more complex like Mail and Calendaring.

Sizing Domino applications is a very complex task due to the varying types of applications as well as the varying ways in which applications can be designed. Because of these variations, each application places quite different demands on system components such as CPU, memory, and disk.

The application discussion concentrates on the interpretation of some of the application concepts discussed in WLE as well as a short discussion on the Domino application rating used to rate the complexity of an application. To accompany the application rating discussion there are also a few questions that may help in determining a rating of an application if it does not compare to any of the examples included in WLE.

3.6.1 Example application comparisons

Today no good tools exist on any Domino platform to simulate the load characteristics of an application and map the performance to a specific configuration for a desired number of users. As stated earlier, the most accurate way to size a system for a specific application is to benchmark the application on a desired platform with the exact resource configuration.

Since that option is not always feasible or available, Workload Estimator directs you to arrive at a *ballpark* estimate of the iSeries capacity you will need by providing examples of several applications and assisting the user in comparing their application to one or more of the examples.

There are currently seven example applications that represent typical Domino applications being deployed as shown in Table 3-2. For a more detailed definition on some of the example Domino applications, go to:

<http://www-1.ibm.com/servers/eserver/iseries/domino/d4apps2.htm>

Table 3-2 User-written application examples

Application example	Rating
Custom (default)	6
QuickPlace Light Publishing	3.5
Web Document Library Application	2.4
Customer Relationship Management	5.5
Web Shopping Application	6
Web Shopping DB2/400 Application	6.3
Domino.Doc (light)	0.5
Domino.Doc (medium)	1.5
Domino.Doc (heavy)	4.5

Once the comparison is made, you make an educated guess at an application rating for your own application. If your application fits closely to one of the specific examples in WLE, choose the appropriate example application from the application type list as shown in Figure 3-5.

The application rating is then selected by WLE, based on which application type was chosen. The application rating found in Workload Estimator is a complexity rating of the entire application. More specifically, the rating is used to determine resource requirement projections for the different applications.

Using the application rating scale

Because there are a wide variety of Domino applications being deployed today, a majority of them are probably not going to compare exactly to the examples described in the Estimator. Therefore, the ratings specified for the example applications in Workload Estimator can be used as references when comparing those applications to your applications.

Once you determine whether your application places more or less demand on the system resources than the example applications, you can select a custom application rating. If you selected the custom application type, it is up to you to specify the application rating. An application rating of 1 would represent a very light application; a rating of 3 would be a moderate application; and something around an 8 would be a fairly heavy application.

IBM Workload Estimator for iSeries - Microsoft Internet Explorer

File Edit View Favorites Tools Help Address <http://www-912.ibm.com/servlet/EstimatorServlet> Go

IBM Workload Estimator for iSeries
an IBM @server

Version: 2001.4
03-Dec-01
www-912

Domino #1
Application Definition

Keep in mind that the performance of Domino applications varies widely.
Please refer to the [help text](#) before continuing with the estimation.

1. Number of [application users](#):

2. Percent of application users active at the same time
([Concurrent](#)):

3. Select the [type](#) of application:
0 11 **2.4**
Will use IBM supplied values.

[Application Rating](#)

4. Type of [application access](#):

5. How would you classify the local [DB2 Universal Database access](#) of the applications?

6. Amount of disk storage needed for shared and/or [application databases](#): (MB)

7. Number of [partitions](#) for Written Applications in this instance of the Domino workload (*CALC will automatically include 0.0 partitions):

So far a total of **0.0** partition(s) are being used by **this** instance of the Domino Workload.
A total of **1** partition(s) are being used for all instances of Domino in this estimation.

Figure 3-5 Workload Estimator Application Definition

As new technology becomes available, Domino applications continue to grow in complexity. For that reason, the application rating scale does not stop at 8. There is room in the scale to rate an application as heavy as 11.

Here is an example of how to use the sample applications to compare with your application and project an appropriate application rating. The first task is to figure out which application most resembles your solution. Once this is completed, there are a couple of ways to proceed:

- If it is determined that your application users will generate similar activities as the sample application, but at a different rate, then use an application rating that is proportionately more or less in the same ratio as the rate difference.

For example, perhaps you are planning to implement a Web shopping application, so that the users are completing a group of transactions similar to the one described for Web Shopper, but are going to be completing the sequence at a rate of 180 seconds instead of 90 seconds. Then this application would use a rating that is approximately half the rating of Web Shopper. Your application is half as heavy as the example.

- It is also possible that the rate and sequence of transactions for the expected workload is quite similar to an example in the Estimator, but the users will be completing heavier or lighter transactions. There is a little more guess work that is required to go into this comparison.

It is important in this example to have a good idea of the complexity of each of the transactions that are completed. It is then up to you to decide how much of a difference the transactions are compared to the example application script. Section 3.7, “Characterizing existing applications” on page 35, includes some questions and metrics that are useful when trying to make the complexity classification of an application.

Users versus HTTP hit rates

When defining a mail or application profile in Workload Estimator, it is necessary to specify a number of active⁵ users. For Web applications, such as the Web Shopping example, it can be difficult to predict the number of concurrently active users.

From an HTTP standpoint, it is a bit easier to discuss *hit rates*. The HTTP protocol does not have the concept of a session and, in many cases, the browser users don’t even need to authenticate. The Web master knows how often a certain URL has been visited, but it is difficult to find out whether 100 hits have been performed by a single person or by 100 different people.

When you use one of the artificial Web applications listed in Table 3-2 on page 31 to rate the complexity of your real application, you also need to specify a number of concurrently active users to determine how often they are called. Those applications are described in detail in the *Domino for AS/400: Application Sizing Examples* white paper at:

<http://www-1.ibm.com/servers/eserver/iseries/domino/d4apps.htm>

The white paper can help you to calculate the active users, if you know only the HTTP hit rate. Table 2, “Web Document Library Estimated Sizing” in the white paper (partially shown here in Table 3-3), allows you to determine a ratio between concurrent users and hits per second by dividing those two numbers. That way you find out that (according to this model application) 1 hit per second is generated by 15 active users, or in other words, 100 active users produce 6.6 hits per second.

For example, assume you consider your own Web application to have the same complexity as the Web Document Library sample application and you expect 120 hits per second. Applying the ration described above would result in 1800 (120 times 15) concurrently active users. You can specify 1800 registered users and concurrency rate of 100% or 3600 registered with 50% active users.

Table 3-3 Web Document Library Estimated Sizing

Model	Processor feature #	N-way #	Concurrent users	Hits per second	Hits per day
270	2422	1	247	16.5	1,422,720
	2423	1	397	26.5	2,286,720
	2424	2	777	51.8	4,475,520

⁵ The Workload Estimator asks first for *registered users*, but what really counts are the active users. This is based on the concurrently active percentage multiplied by the number of registered users.

Model	Processor feature #	N-way #	Concurrent users	Hits per second	Hits per day
820	2425	1	397	26.4	2,286,720

This means that (concurrently active) users generate various numbers of hits during an active session with the application. Hit rate values, along with the equivalent number of concurrent users corresponding to those hit rates, are included in the definition and description of the Web applications in the white paper.

Domino.Doc now part of the Domino application profile

Note, Domino.Doc is now included in the Domino application profile. In older versions of the WLE, there was a separate profile for Domino.Doc. If you are used to using the Domino.Doc profile of the Domino workload, it will be simple to transform the information into the Application profile.

There were originally three different complexities of the Domino.Doc application, that had different rates in which the script was completed. These three profiles are still included in the Estimator. They are now application-type examples to be used separately or to be used for comparisons with other custom written applications.

If you are used to combining complexities of your Domino.Doc users with the complexity sliding bar, there are two ways to represent this in the Estimator. The easiest and most accurate way is to define two or three separate Domino workloads and define the different Domino.Doc user complexity groups each in a separate workload.

It is also possible to make some estimates at what the average complexity your users would be and use only one Domino workload. This way introduces another level of uncertainty and should be used with caution.

3.6.2 Database capacity

In this context, *database* refers to a DB2 UDB database and its associated processes, not a Notes database. Only the time spent in DB2 database code as well as access methods, such as SQL and Query/400, is considered to be DB2 database processing. The majority of the processing in these methods is during the accessing of database records through SQL, ODBC, JDBC, Query/400, DRDA, and other database access methods.

One specific case in WLE when this metric is important is when sizing a V5R1 Dedicated Server for Domino iSeries model. For an in-depth discussion on the V5R1 Dedicated Server for Domino, refer to the white paper at:

<http://www.ibm.com/eserver/iseries/domino/dsd.htm>

The reason for discussing the database capacity as part of the general Domino sizing concepts of WLE is because of the way it is now used in V5R1 of OS/400 for Dedicated Server for Domino systems. The V5R1 Dedicated Server for Domino logic allows for complementary workloads to be deployed as long as certain criteria are met. To account for this, the Estimator now allows for complementary workloads to be projected onto a Dedicated Server for Domino, but with these requirements in place:

- ▶ Domino must be the majority of the projected total work defined in WLE.
- ▶ Projected DB2 UDB utilization is less than the rated DB2 capacity for the model.

The rated DB2 capacity is equal to 15% of the CPU for Dedicated Server for Domino models. For more information on optimizing an application's use of DB2 database processing, refer to the iSeries Information Center for V5R1 at:

<http://publib.boulder.ibm.com/html/as400/v5r1/ic2924/index.htm>

3.7 Characterizing existing applications

Because of the complexity and variety in the Domino applications being deployed, here are some questions and characteristics that would be useful to classify an application that was not comparable with any of the examples found in WLE:

- ▶ Where is the accessed data located?
 - In small databases?
 - In a large number of databases?
 - In DB2 databases?
- ▶ How is the data structured for accessibility?
- ▶ How is the load distributed on the server?

If the answers to these questions lead to an observation that your solution is quite complex, it is appropriate to suggest a fairly high application rating. The following sections offer some examples of design metrics that may have an impact on the server load or capacity as well as user response times and can help answer the questions above.

Some application characteristics do not necessarily add extra load to the server that, in turn, affects the total performance, but have an affect on the user response time.

Too many views or agents

One metric that could potentially have quite an impact on server performance if not tuned appropriately is the number of views. If your application has hundreds of views and is constantly refreshing them, the application could prove to be heavier than previously noted.

Similarly, the number of agents in an application could cause a significant performance hit. The number of add-in tasks that are active in the application also has a significant impact on the server load.

DB2 access and network overhead

Another factor that could impact the actual load put on the server by an application is the extent to which the application accesses backend data like DB2. The more network layers and connections that are involved in accessing the data, the bigger the potential is for longer response time.

After you determine if the application has any of these characteristics, and to what extent, you can now select an appropriate application rating. There is no exact science to calculate an application rating based on the design components of an application. However, having knowledge of the metrics discussed here will help you to determine the appropriate input values when using WLE.

Applications currently running on an other platform

If you currently have a Domino application today running on a platform other than iSeries, there are a couple of ways to obtain useful information needed to define the profile of the Domino solution in WLE. As mentioned previously 3.2, “Benchmarks to determine the load of specific workloads” on page 21, it is possible to use the published NotesBench results from a different platform to help characterize and define a specific workload for WLE.

3.8 Additional options for more accurate estimation

Aside from the actual Workload definitions, there are additional input capable parameters in the tool that give the user options to direct the recommended system. To access these additional parameters, go to the Navigation bar found in the top left corner of every WLE screen. Choose **Edit** followed by **Options**. This opens up the Options page (Figure 3-6) where you can adjust assumptions that apply to the entire iSeries environment that you are estimating.

Figure 3-6 Input-capable parameters in WLE

These options affect both the size of the projected workload or workloads and the manner in which the Estimator combines multiple workloads. Here are some descriptions to help aid in using these options to guide the Workload Estimator in recommending an appropriate system for your solution.

Target Processor Utilization

This parameter affects how the tool selects systems. The Estimator attempts to select a system that will provide processor utilization no greater than the specified *Target Processor Utilization*. To select a system that will provide adequate room for incidental growth as well as base operating system resources, the Workload Estimator default is 70%.

You may change this value based on specific needs. However, we do not recommend setting the default utilization above 70%. This could result in performance problems during peak usage times, when there are “spikes” in the system utilization. It is important to remember that, when you adjust this option, the recommendations of the WLE are *estimates*. This means that some range should be allowed for around the projected CPU Utilization.

Target Interactive Feature Utilization

The Estimator attempts to select a system whose interactive feature will be used by less than or equal to the specified amount. The default for this parameter is 100%.

Disk Storage Percent Full

For each workload defined, the Estimator calculates the minimum amount of disk storage that will fulfill the workload requirements. Before you select a system for the estimation, the Workload Estimator adds together the storage for each workload, plus extra storage requirements for base operating system requirements.

If you want the Estimator to calculate extra disk storage as a percentage of the total required, you can lower this parameter. This is because the parameter is defaulted to 100%. This means that the Estimator assumes that the workload or workloads can consume all of the disk storage available for the selected system.

If you lower this value, for example to 75%, then the Estimator would “save” 25% of the disk storage available to the system and only allow the workloads to consume 75% of the storage available. This would be an appropriate option to change if you knew that, for example, you were sizing a mail workload and your users' mail databases were projected to grow in size.

Disk Storage Type

The 10K RPM drives have a faster latency than the 7K RPM drives. This means that they have a faster spinning rate for reading and writing to disk. Potentially, the 10K drives could result in approximately ten to fifteen percent more throughput with most workloads.

Target Family

The new iSeries servers with the new, faster processors result in better performance over the previous 170, 720, 730, and 740 models. The Workload Estimator will default to the highest available technology. Unless there is a specific case in which you want to size your solution on a previous technology system, we suggest that you leave this parameter with the default.

3.9 Using IBM Workload Estimator for iSeries results

The Selected System that is recommended by WLE is a processor model that will support the processing, disk, and memory requirements calculated from the questions answered in each of the workloads defined. All of the systems capable of supporting the work defined are identified and reported in the Selected System drop-down list as shown in Figure 3-7.

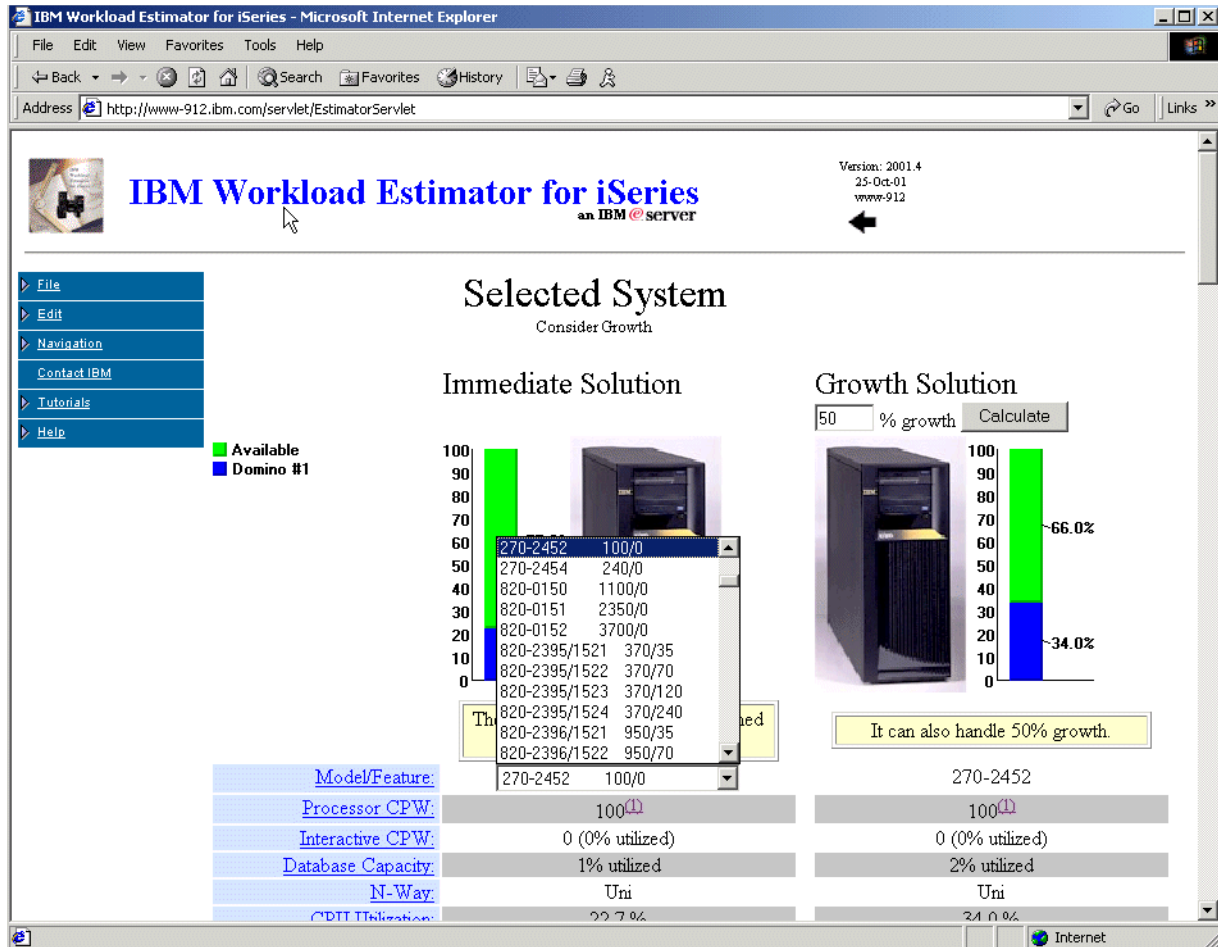


Figure 3-7 Selected Systems page

The smallest system model capable of supporting the resources required for the workload or workloads that is defined, as well as meeting the criteria specified in 3.8, “Additional options for more accurate estimation” on page 36, is presented as the recommended system.

The Selected System panel actually includes the specific processor model, interactive feature, processor Commercial Processing Workload (CPW), interactive CPW, database capacity, processor utilization, memory, and the required number of disks and disk capacity.

Through the options described in 3.8, “Additional options for more accurate estimation” on page 36, the user can modify how the WLE arrives at these recommendations for customizing. One suggestion is to lower the Target Processor Utilization. This would allow for projecting growth. It would also allow for spikes in the system utilization during peak hours of operation.

For specific cases on the opposite end of the spectrum, if you know that yours is very constant and there are no major peak times, it is possible to set the target utilizations a bit higher. After you set the appropriate options, it is time to recommend a system that is capable of deploying the defined solution. There are a couple of resources that are reported by the Workload Estimator that need a bit of explanation on how they refer to a Domino solution.

Processor CPW

The Processor CPW, reported in the results of the Workload Estimator (Figure 3-8), should not be treated as the total CPW in all cases. In some instances, such as the Dedicated Server for Domino, this metric is actually the non-Domino processor CPW. For a more detailed discussion on the Dedicated Server for Domino models and the non-Domino processor CPW, see the Dedicated Server for Domino white paper “Enhanced V5R1 Processing Capability for the iSeries Dedicated Server for Domino”, which is available on the Web at:

<http://www-1.ibm.com/servers/eserver/iseries/domino/pdf/dsdjavav5r1.pdf>

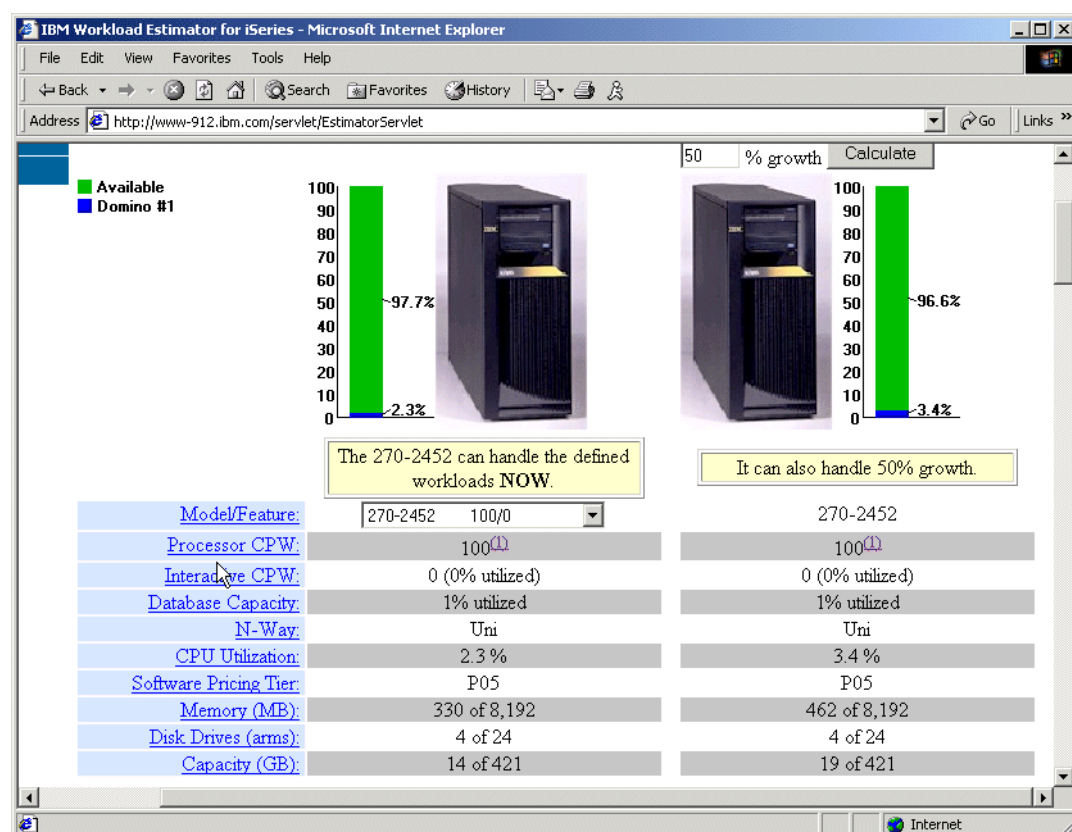


Figure 3-8 Processor CPW

Database Capacity

The *Database Capacity* metric reported for the Selected System shown in Figure 3-9 is a measure of the percentage of the overall CPU that is used to perform the DB2 database (not Domino database) processing. Database processing generally includes processing time in SQL, ODBC, JDBC, Query/400, DRDA, and other database access methods.

If the database CPU is fully consumed, this does not mean that all of the capacity on the system is being utilized. It is possible to add more work other than database processing to take advantage of the remaining cycles.

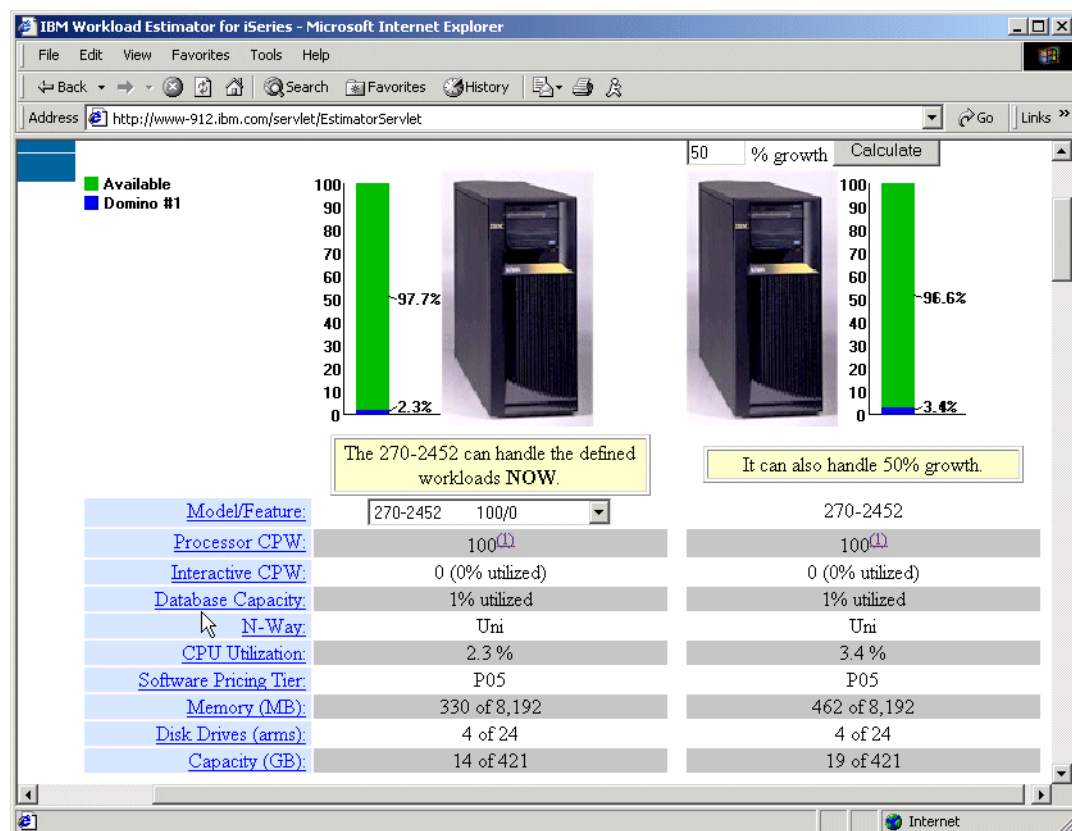


Figure 3-9 Database Capacity

Disk Drives

The *Disk Drives (arms)* value (Figure 3-10) is the number of disk drives required to support the defined workload. This number is determined by the speed of the drives and the number of disk operations per second necessary to do the work. It does not take into account the disk drive size available. You should always configure at least this number of disk drives, even though you do not need that much capacity.

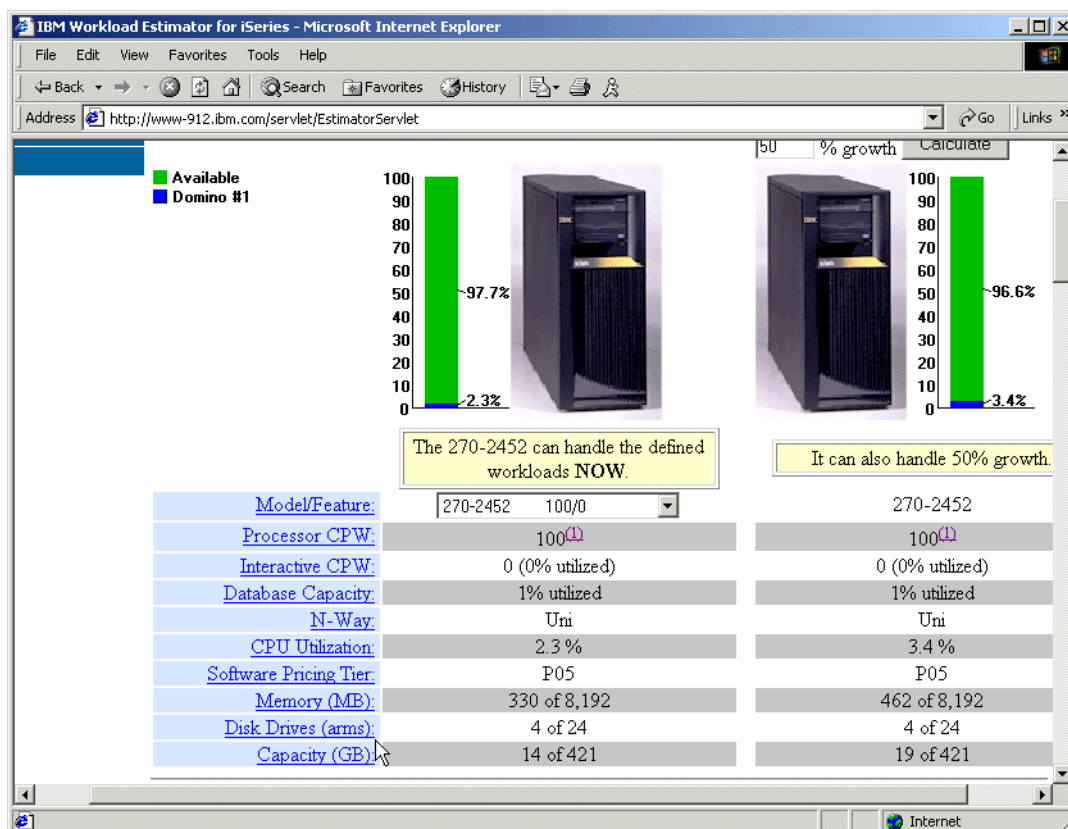


Figure 3-10 Disk Drives (arms)

Capacity (GB)

The Capacity shown in Figure 3-11 is a separate calculation based on the amount of disk storage defined for the mail databases plus the storage needed for any application databases. The calculations for disk capacity and the number of disk drives are done separately. Only taking one into consideration could result in a number that is too low for the other. For example, by selecting large disk drives to satisfy the capacity requirements of WLE, there may not be enough arms for the best performance.

It is also worth noting that the value reported for the Selected System requirements is a minimum requirement. By including more drives, the system will have better parallel access to the data, which would reduce the data access time significantly.

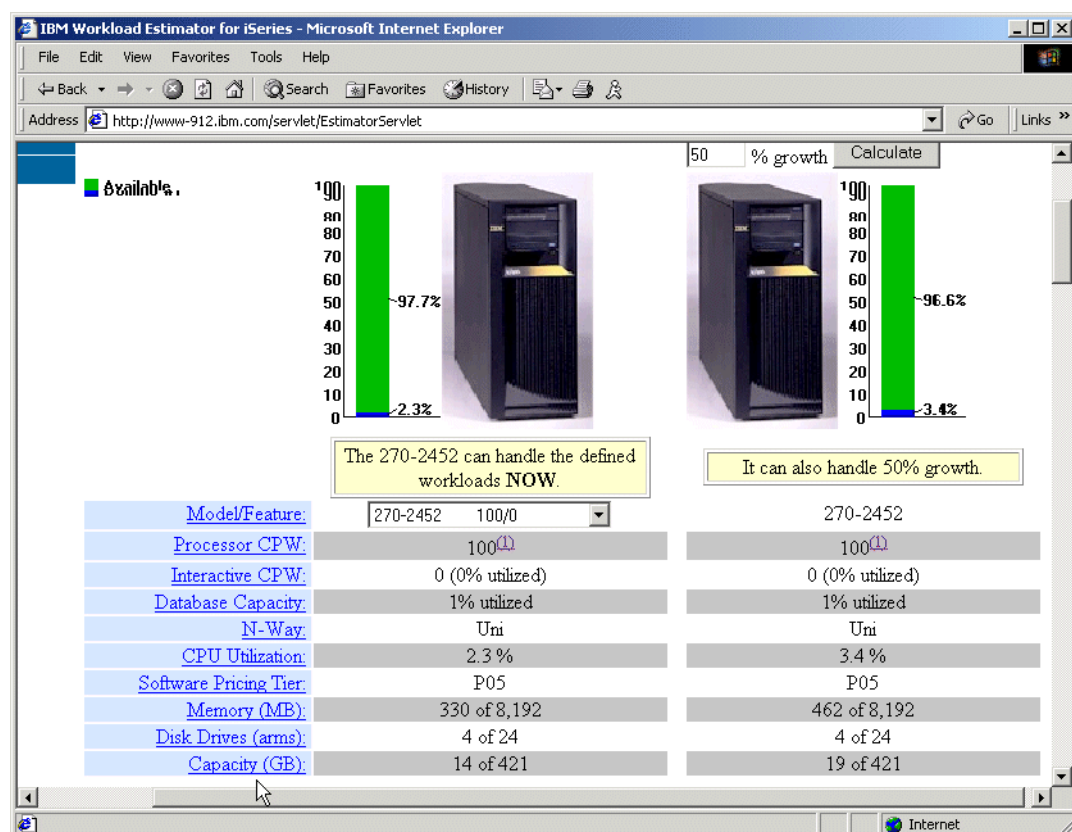


Figure 3-11 Capacity (GB)

Growth Solution

Growth Solution (Figure 3-12) is also included in the Workload Estimator Results. It is selected by applying the growth rate, defaulted to 50%, to all of the system resources used to determine the initial recommended system. It is possible, if the Immediate Solution is capable of supporting 50% more resource, to have a Growth Solution be the same system as the Immediate Solution.

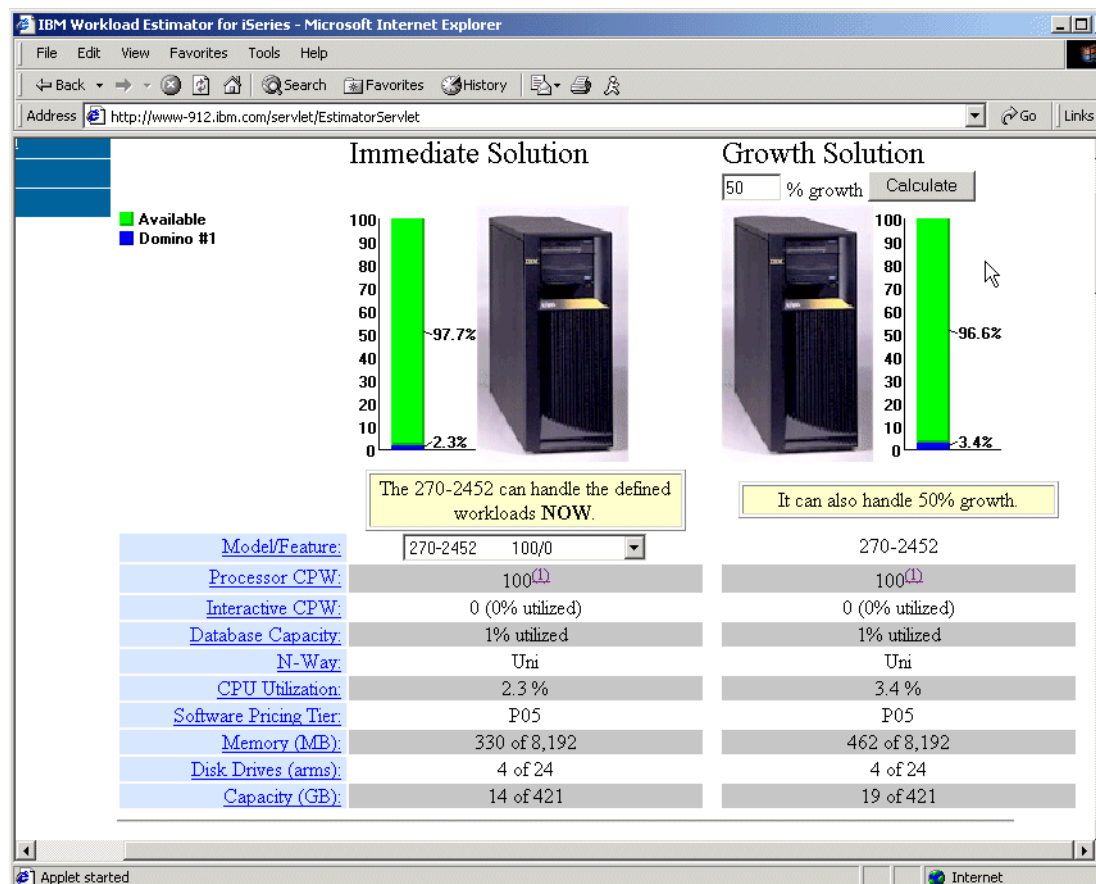


Figure 3-12 Growth Solution

Further information on both the Immediate and Growth Solutions is summarized in Figure 3-13.

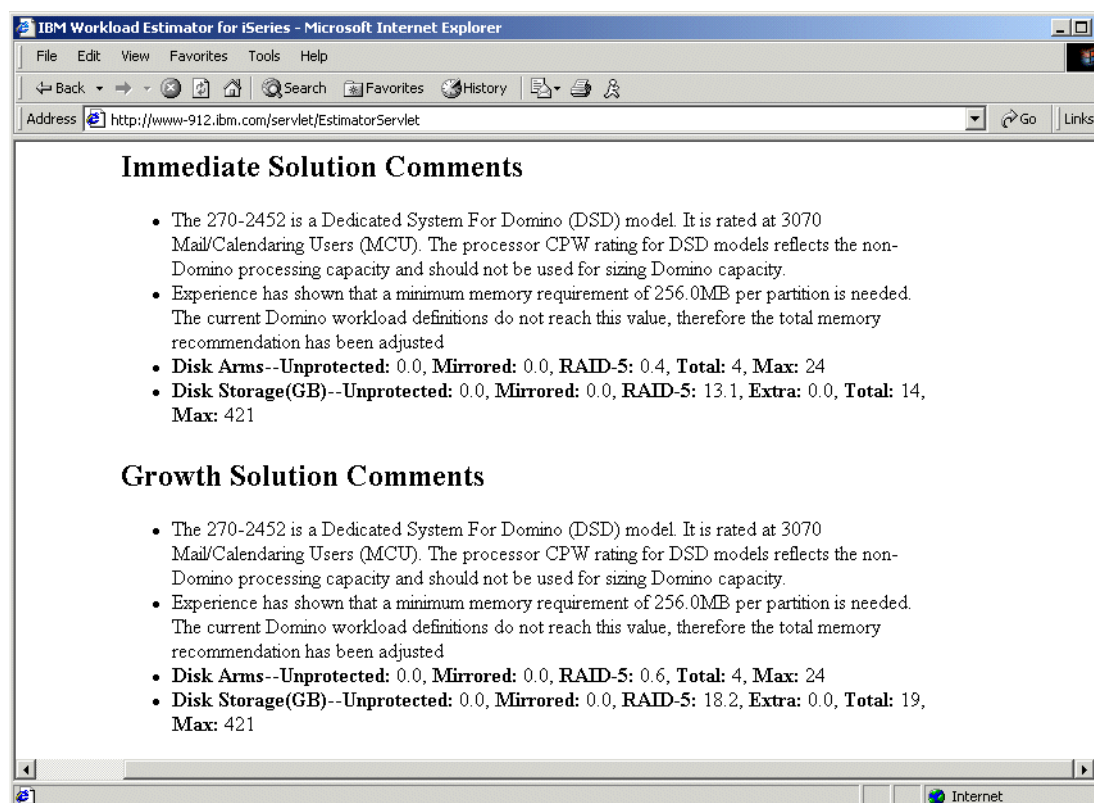


Figure 3-13 Immediate and Growth Solutions

3.10 Additional tips and insights

The Estimator help text for the Domino workload is very helpful in answering questions about the basic information on the workload. It is impossible to cover all of the Domino material that is used in sizing and performance measurements necessary. This section covers a few of the items that are not found in detail in the WLE help text, but that should be useful when using the WLE to help size a Domino solution.

Dedicated Server for Domino

For specific Dedicated Server for Domino information, see Chapter 12, "The iSeries Dedicated Server for Domino" on page 341. You should also see the Dedicated Server for Domino white paper "Enhanced V5R1 Processing Capability for the iSeries Dedicated Server for Domino", which is available on the Web at:

<http://www-1.ibm.com/servers/eserver/iseries/domino/pdf/dsdjavav5r1.pdf>

CPW versus CIW

The Commercial Processing Workload rating of a system is determined based on measurements using a synthetical workload and not based on a real existing application. CPW is designed to evaluate the system and associated software in a commercial environment. It is not representative of any specific environment, but generally applies to the commercial computing environment.

The applications that follow the CPW patterns tend to have many jobs running short transactions using relatively few CPU instructions and spending much time waiting for I/O to complete. Usually this is an environment where more time is spent by system code performing DB2 operations, rather than by the application executing processor instructions.

CPW ratings can be used best when the application you are sizing is expected to be used for traditional business uses such as order entry, billing, etc. Domino applications are not sized well using CPW ratings, because typically the ratio of processing time to I/O time for Domino is very different from CPW.

When sizing a Domino application, it is more important to consider the necessary CPU cycles per user per time versus the time waiting for I/O. For this reason, the *Compute Intensive Workload* (CIW) metric has been introduced for newer iSeries models to improve Domino sizing projections. CIW ratings are used when most of the work by the application is done within the application code instead of system services. For example, a Domino Mail and Calendar workload might spend 80% of the total processing time outside of OS/400, while the CPW workload spends most of its time in OS/400 database code.

Since the instruction path length tends to be longer than CPW-like workloads, it is very likely that processors will spend less time switching from task to task. Therefore, it is not as important to have a big cache. The cache affect is also due in part to the data being accessed. Because CPW does more I/O, the cache plays a bigger role than it does for CIW.

Some L2 cache *is* very important for Domino, especially for applications. That is why we always recommend that you choose those iSeries or AS/400 models for Domino that have L2 cache (see Table 3-4 on page 48 and Table 3-5 on page 48). But Domino applications that are more CPU-intensive and more comparable to CIW benefit more from additional processor speed rather than a bigger cache. Also, because CIW-like applications spend more time on individual processors, they tend to scale better on multiprocessor systems, than mail and CPW does.

You can find further information on CIW in Appendix A of *iSeries Performance Capabilities Reference Version 5 Release 1*, SC41-0607. You can also find CPW and CIW ratings for iSeries servers in Appendix D of the same manual.

Typical, Simple, or Mail and Calendaring Users

When using Workload Estimator to size for a Domino mail solution, it is important to understand the relationships between these three metrics:

- ▶ Simple Mail Users (SMU)
- ▶ Mail and Calendaring Users
- ▶ Typical Mail Users

Due to the NotesBench audit rules, we are not allowed to publish results generated by NotesBench workloads without having them officially audited. Because it is not possible to audit results for every possible server configuration, the SMU and MCU metrics are used.

- ▶ *Simple Mail Users*
 - Open mail database that contains documents that are 1 KB in size
 - Open the current view
 - Open five documents in the mail file
 - Categorize two of the documents
 - Compose two new mail memos 1 KB in size (every 90 minutes)
 - Mark several documents for deletion
 - Delete documents marked for deletion
 - Close the view

► *Mail and Calendaring Users*

- Open the mail database that contains documents that are 10 KB in size
- Open the current view
- Open five documents in the mail file
- Categorize two of the documents
- Compose two new mail memos 10 KB in size (every 90 minutes)
- Mark several documents for deletion
- Delete documents marked for deletion
- Create one appointment (every 90 minutes)
- Schedule one meeting invitation (every 90 minutes)
- Close the view

We decided that these two scripts do not represent a typical mail workload. Therefore, we have determined that a *typical* user is three times as heavy as a SMU and twice as heavy as an MCU. This means that a given server that can support x number of Typical users can support 3x SMU and 2x MCU. To translate these metrics into Workload Estimator terminology, the equivalent of a Typical user is a *moderate* mail user.

Registered users versus active users

Note, that so far we were only talking about *active users*. Most people asking for the proper size of a Domino server will start similarly to: “We will have xx Domino users” The number xx includes typically all users who will use the Domino server at some point in time. In the terminology of the Workload Estimator, these are called *registered users*.

The number of registered users is the first question to be answered for each WLE workload. However, this number alone cannot be used as a base for sizing, unless you respond to the following questions. The first question and probably most important questions is: “What is the percentage of *concurrently active users*?” Multiplying that value with the registered users brings you the number of *active* users.

For example, the Model 820-2457 is rated at 6660 MCU at 70% CPU. If you specify 6660 users in Workload Estimator⁶, the Estimator will project a utilization of 69.9% CPU utilization of the 820-2457 at 50% concurrency. If the concurrency rating is changed to 100%, the number of users projected at 70% is 3330, or one half of the MCU number.

3.11 Consolidating Domino servers from other platforms

Many times we hear a question like this: “How does an iSeries (or AS/400) Model xxx compare to server yyy running operating system zzz with a processor of nn MHz clock speed?” Most often this question comes up when someone plans to consolidate multiple servers from a PC or UNIX platform to one or more iSeries servers. The correct answer is always: “It depends!”.

It depends on:

- The *complexity* and type of Domino application you are planning to run, as detailed in the previous sections.
- The *entire server configuration*. That is, processor clock speed (“MHz”) can only set an upper limit of the potential capabilities at most. Many other factors play an important role and can further limit the capabilities if they are not sized appropriately, such as size of memory, the number (not size) and speed of disk drives, and the capacity of communication lines.

⁶ Not selecting any additional options such as clustering or other applications.

While these components can be set by the person who configures the hardware, there are other factors depending on the internal architecture and design of the server. Such factors include: bus speed, internal data transfer band width, multitasking overhead, and others.

Having said that, what can you do to estimate the appropriate size of an iSeries server, when you already know what server is needed on a different platform? Similar to what we said for sizing a completely new server, you can use benchmark data to provide an order of magnitude, as long as you are aware of the limitations of this method.

How important is the processor clock rate (MHz)?

Due to the dramatic developments in the area of processors during the last decade, it seems to many people that the clock rate of the processor (MHz) appears as the one and only indication of the speed of a computer system. However, the processor is the most important component and is not alone responsible for performance.

There are some facts that describe where the processor speed is more or less important:

- ▶ Capacity is affected by a complicated combination of processor, I/O, and overall efficiency factors. All applications need a different proportion of these resources. That's why we use CPW, CIW, MCU, or other kinds of well-defined workloads to classify applications.
- ▶ MHz primarily affects response time of CPU intensive transactions. Some applications are more processor intensive than others. More importantly, however, is the fact that a short response time measured for a single transaction does not indicate at all how the server will be able to handle hundreds or thousands of users.
- ▶ On servers with multiple processors, "adding up the MHz" will underestimate the capacity of iSeries servers and probably overestimate the capacity of servers on some other platforms due the relative efficiencies of the architectures.
- ▶ When comparing to existing servers, it is necessary to know at which CPU utilization they work and take it into account. You can use the Domino platform statistics (see 5.1.4, "Platform-dependent statistics" on page 137) to find out how the utilization of the processor can raise during peak times.

What about CPW?

The Commercial Processing Workload should not be used to estimate for three main reasons:

- ▶ Most, if not all, Domino applications show a different behavior than the one defined in CPW.
- ▶ The CPW published for the Dedicated Server for Domino applies only to the non-Domino workload running on a Dedicated Server for Domino. See 12.2, "Dedicated Server performance behavior and capacity" on page 342, for more information.
- ▶ CPW is normally not published for any non-iSeries platform.

Simple Mail Users or Mail and Calendaring Users

If the existing servers have a NotesBench audit published, you can roughly compare the Simple Mail Users with the NotesBench R4.5 Mail users and the Mail and Calendaring Users with NotesBench R5 Mail and Calendaring Users. While this method is probably the most accurate, there still some restrictions to consider:

- ▶ The term *users* in SMU, MCU, and NotesBench users should be considered as a *benchmark unit* rather than real users. The typical mail users, as described in "Typical, Simple, or Mail and Calendaring Users" on page 45, may come close to reality in a mail-only environment, but should never be used to estimate the workload for other Domino applications.

- ▶ Many Domino applications tend to have a more CPU intensive workload, which is more comparable to CIW as it is to mail users.
- ▶ The benchmarks assume a balanced configuration, that is an appropriate proportion of CPU, memory, and disk resources, which may not be true for an existing server.
- ▶ When comparing to existing servers, their resources may not be fully utilized.

When keeping the above points in mind, comparing the NotesBench and MCU ratings of existing servers is still the most reliable method to estimate the appropriate configuration of an iSeries server. In addition the CPU clock rating, together with actual CPU rating can be used to verify the necessary number and type of iSeries processors.

In any case, the Workload Estimator should also be used to ensure a balanced system, as described in the previous sections or this chapter.

The following sections should help you to gather some information about the different iSeries models and compare it to benchmark data published by the NotesBench consortium on the Web at: <http://www.notesbench.org>

3.11.1 Capabilities of various iSeries and AS/400 models

Table 3-4 and Table 3-5 on page 48 show the Mail and Calendaring Users (estimated at 70% CPU capacity) for the those 270 8xx Models announced in 2001. The other tables on pages 49 through 50 show the SMU and MCU ratings of older iSeries and AS/400 models. You may use the tables to compare the potential capacities of the different iSeries models.

Although the ratings Mail and Calendaring Users and Simple Mail Users are not officially the same as the NotesBench ratings, the tables can give you a very rough idea to compare the capabilities of a certain iSeries model with another server which does have published NotesBench ratings.

Table 3-4 and Table 3-5 describe performance capabilities of the current (as of the year 2001) iSeries models as they relate to MCU.

Table 3-4 iSeries Dedicated Server for Domino performance details

Model	Proc. feature	CPUs	Chip speed	L2 Cache	MCU rating	Typical mail
820	2458	4	600 MHz	4 MB	11,810	5,905
	2457	2	600 MHz	4 MB	6,660	3,330
	2456	1	600 MHz	2 MB	3,110	1,555
270	2454	2	600 MHz	4 MB	6,660	3,330
	2452	1	540 MHz	2 MB	3,070	1,535

Table 3-5 Further iSeries servers

Model	Proc. feature	CPUs	Chip speed	L2 Cache	MCU rating	Typical mail
840	2461	24	600 MHz	16MB	77,800	38,900
	2420	24	500 MHz	8 MB	55,610	27,805
	2418	12	500 MHz	8 MB	28,430	24,215
830	2403	8	540 MHz	4 MB	22,900	11,450

Model	Proc. feature	CPUs	Chip speed	L2 Cache	MCU rating	Typical mail
	2402	4	540 MHz	4 MB	10,680	5,340
	2400	2	400 MHz	2 MB	4,490	2,245
820	2438	4	600 MHz	4 MB	11,810	5,905
	2437	2	600 MHz	4 MB	6,660	3,330
	2436	1	600 MHz	2 MB	3,110	1,555
270	2434	2	600 MHz	4 MB	6,660	3,330
	2432	1	540 MHz	2 MB	3,070	1,535

Table 3-6 shows Domino mail workloads for the previous generation of the 270 and 8xx models. MCU results are based on 70% CPU utilization and should not be compared to the audited NotesBench numbers.

Table 3-6 Domino workloads on first generation of 270 and 8xx models

Model	Proc. feature	CPUs	Chip speed	L2 Cache	MCU rating	Typical Mail	Simple mail
840	2420	24	500 MHz	8 MB	55,610	27,810	83,420
840	2418	12	500 MHz	8 MB	28,430	14,220	42,650
830	2403	8	540 MHz	4 MB	20,910	10,460	31,370
830	2402	4	540 MHz	4 MB	10,680	5,340	16,020
830	2400	2	540 MHz	2 MB	4,490	2,250	6,060
820	2398	4	500 MHz	4 MB	9,890	4,950	14,840
820	2397	2	500 MHz	4 MB	5,610	2,810	8,420
820	2396	1	450 MHz	2 MB	2,570	1,290	3,860
820	2395	1	400 MHz	-	1,650	830	2,480
270	2253	2	450 MHz	4 MB	5,050	2,530	7,580
270	2252	1	450 MHz	2 MB	2,570	1,290	3,860
270	2250	1	400 MHz	-	1,600	800	2,400
270	2248	1	400 MHz	-	810	410	1,220

Table 3-7 shows Domino mail workloads for 170 and 7xx models.

Table 3-7 Mail workloads for 170 and 7xx models

Model	Proc. feature	CPUs	Chip speed	L2 Cache	MCU rating	Typical mail	Simple mail
740	2070	12	262 MHz	4 MB	17,860	8,930	26,790
740	2069	8	262 MHz	4 MB	12,350	6,180	18,530
730	2068	8	262 MHz	4 MB	10,960	5,480	16,440

Model	Proc. feature	CPUs	Chip speed	L2 Cache	MCU rating	Typical mail	Simple mail
730	2067	4	262 MHz	4 MB	6,900	3,450	10,350
730	2066	2	262 MHz	4 MB	3,530	1,760	5,290
730	2065	1	262 MHz	4 MB	18,70	940	2,810
720	2064	4	255 MHz	4 MB	4,920	2,460	7,380
720	2063	2	200 MHz	4 MB	2,570	1280	3,850
720	2062	1	200 MHz	4 MB	1,340	670	2,010
720	2061	1	200 MHz	-	940	470	1,410
170	2388	2	255 MHz	4 MB	3,150	1580	4730
170	2385	1	252 MHz	4 MB	1,550	780	2,330
170	2292	1	200 MHz	-	940	470	1,410
170	2291	1	200 MHz	-	450	220	670
170	2290	1	200 MHz	-	280	140	420
170	2289	1	200 MHz	-	190	100	290

Table 3-8 shows the performance capabilities on 6xx and Sxx models. These results are at 70% CPU utilization and should not be compared to the audited NotesBench results. Use these figures when comparing AS/400 systems only.

Table 3-8 Simple mail users on 6xx and Sxx models

Model	Proc. feature	CPUs	Typical mail	Simple mail
600	2129	1	33	100
	2134	1	43	130
	2135	1	66	200
	2136	1	116	350
S10	2118	1	66	200
	2119	1	116	350
620	2179	1	123	370
	2180	1	166	500
	2181	1	300	900
	2182	2	666	2000
S20	2161	1	166	500
	2163	1	300	900
	2165	2	666	2,000
	2166	4	1,133	3,400
640	2237	1	466	1,400

Model	Proc. feature	CPUs	Typical mail	Simple mail
	2238	2	800	2,400
	2239	4	1,533	4,600
S30	2257	1	466	1,400
	2258	2	800	2,400
	2259	4	1,533	4,600
650	2240	8	2,900	8,700
	2188	8	5,500	16,500
	2243	12	4,166	12,500
	2189	12	7,966	23,900
S40	2207	8	5,500	16,500
	2261	12	4,166	12,500
	228	12	7,966	23,900

3.12 Additional resources

The following Web sites offer you additional resources in sizing Domino for iSeries:

- ▶ Dedicated Server for Domino white papers:
<http://www.ibm.com/eserver/series/domino/dsd.htm>
- ▶ Application sizing examples:
<http://www-1.ibm.com/servers/eserver/series/domino/d4apps.htm>
- ▶ NotesBench consortium: <http://www.notesbench.org>



Basic concepts of performance analysis

This chapter provides recommendations on evaluating Domino performance. The primary tool used for collecting performance data on iSeries is *Collection Services* (previously called Performance Monitor). If a performance problem cannot be identified by the data collected with Collection Services, the next tool to use is the *Performance Explorer*. The collection functions and related commands in Collection Services and Performance Explorer are part of the OS/400 operating system.

The reporting and analyzing functions for Collection Services and Performance Explorer are part of Performance Tools for AS/400 Licensed Program Product (5769-PT1). Performance Tools is a separate product that must be purchased. Its cost is based on the system model. Contact your sales representative for more information.

An iSeries server can be viewed as a stable, well-balanced system where every job has a slot (run priority) of its own. System jobs in the iSeries server support all basic functions required by the applications. All Domino tasks run as high-priority batch jobs. Some have characteristics that are comparable to traditional (5250-based) interactive jobs. Changing a run priority of a job, without careful planning, may work for some time. In the long run, it may cause unpredictable results. Therefore, you should not add applications without proper preparation.

As with any performance related activity, there are some general considerations that need to be applied when doing performance tuning for Domino on the iSeries server. Performance evaluation and management for Domino for iSeries is similar to performance tuning on the iSeries server. However, special considerations are given to Domino itself and to its performance management.

4.1 Defining a Domino for iSeries performance methodology

Anytime you enter a performance-oriented work environment, you need to consider some general guidelines or what you may call a “high-level view”. To help with this view, you need to:

- ▶ Develop objectives such as customer expectations
- ▶ Define short-term and long-term goals
- ▶ Implement a good strategy for deployment of your objectives
- ▶ Set a benchmark for evaluation of your level of progress

When we talk about objectives, we refer to a clear definition of what the customer expects in a manner that can be measured. It is very important that the customer clearly understands their performance objectives before the development of your performance management strategy. Performance analysis can be a long process. A well-defined strategy can make a difference in the degree of your success. Performance management generally consists of performance measurements and a trend analysis. Some type of performance problem may already exist that may have warranted your involvement.

Setting up a performance management solution for Domino on the iSeries server is more complex than the traditional iSeries performance tuning. You first need some understanding of Lotus Domino. This is essential to understanding the relationship between the processes that are taking place in Domino and how these processes affect the performance of the iSeries server. Because only limited tools are currently available to provide the analysis of Domino for iSeries, you may be required to gather some of the necessary information in making your performance determinations. This redbook was developed to assist you with that process.

This redbook refers to iSeries performance measurement components such as Queuing Multiplier, CPU Service Time Equation, and so on. If you require a better understanding of the components of iSeries Performance Management, we recommend that you see *AS/400 Performance Management*, SG24-4735, and *Performance Tools for iSeries Version 5*, SC41-5340. The best practice for understanding Domino Performance is to have a good understanding of the information in the Domino 5 Administration Help (help5_admin.nsf) database and the iSeries specific *Domino for AS/400 Help* (as400hlp.nsf). Lotus Domino System Administration classes offered by Lotus Education are also available. The majority of the performance tuning parameters available for Lotus Domino is found in detail in the handbook or taught in a System Administration class.

When gathering performance information on Domino on the iSeries server, basic iSeries techniques, in conjunction with some additional gathering of Domino information, provide a good foundation for making performance management decisions.

Defining performance gathering objectives

Consider the following points when collecting performance data:

- ▶ Collect performance data against a set of agreed-to defined objectives. These objectives set goals and expectations and should focus on specific performance issues.
- ▶ Plan on meeting one objective at a time. You may start out with an objective of end-user response time and be asked to pursue optimal transactional throughput. There is a direct relation between your specific objective and the way that the data is collected for that objective.
- ▶ Select the criteria for data collection that is a direct result of the performance objective definition. It should include such things as duration, type of information to be collected, and so on.

- ▶ Select measured data showing a medium-to-heavy workload that the customer feels accurately reflects the typical workload.
- ▶ Define a peak period (or critical applications) to measure.
- ▶ Ensure measurements are made under similar conditions. That is, the same workload, number of active workstations, and so on must be valid for purposes of comparison.

Making valid performance data evaluations and recommendations

Consider the following points when evaluating the data:

- ▶ Consider how performance changes may affect the rest of the business organization such as different user groups.
- ▶ Involve the necessary decision making individuals with discussion-based feedback on issues such as cost, objective clarification, and time schedules.
- ▶ Evaluate business growth on additional resources. Performance information that you gather now is critical in trend analysis of future system requirements.
- ▶ Evaluate the pattern of your system workload. Know whether it is steady, increasing, or decreasing, and what percentage of utilization you are using on your system.
- ▶ Take several measurements before reaching a conclusion. You need a comparative measurement to find acceptable or unacceptable system performance.

Implementing performance improvement changes

Consider the following points when implementing performance changes:

- ▶ Give any change that is made ample time under various working conditions. Managing system performance is a continuous process involving system changes that may not produce an immediate performance improvement.
- ▶ Make and measure changes in a controlled environment.
- ▶ Make only one change at a time. This allows you to gauge the impact of each change.

4.1.1 Relationship between Domino and the iSeries server

Under OS/400, Domino is considered an application that runs within its own subsystem. However, the ramification of putting Domino on your iSeries server can be far reaching. Because Domino has its own server, it has the capability of generating tasks that can compete for system resources with limited administrator control. Domino, by default, runs everything at the same priority. Run priority is further described in 4.9.1, “Run priority” on page 64. This can have a major impact on an iSeries server that has to share resources between Domino and additional business applications. Domino application developers can create complex workflow or other GUI-based client/server applications. They can also route information automatically and exchange information with enterprise software such as relational databases and transaction processing systems.

Today, by adding Domino, the iSeries server enters into a more complex, collaborative environment. This environment requires performance specialists to consider processes that may require looking beyond standard performance techniques. As you come to understand, Domino acts as its own entity on the iSeries server, running Domino transactions to meet the Domino server requirements as well as the requirements of the users. Transactions are also run for any external system requirements such as requests from the OS/400 operating system. Because Domino is so diverse, Domino on the iSeries server can be optimized, with proper planning, to take advantage of OS/400 assets such as scalability. The iSeries server

can also be optimized to take advantage of Lotus Domino assets such as database clustering. Using the loopback option to cluster Lotus Domino databases on a single iSeries server provides an example of efficiency and high availability that is unmatched on other platforms.

The main function of the Domino server is to service its users by providing database access, mail routing, and so on. Frequently, these tasks require interaction with the OS/400 operating system, such as the AnyMail FrameWork for routing messages to the Internet or when Domino users access DB2 UDB for iSeries databases directly through Domino. Although these and other interfaces to the OS/400 operating system use system resources, the Domino server tasks themselves should be the primary focus of performance considerations.

If you understand some of the components of Domino, you should understand the advantages that Domino can provide under OS/400, as well as the performance implications. Some of the Domino components are:

- ▶ Transactions
- ▶ Domino objects
- ▶ Domino base function definitions

4.2 Understanding Domino on the iSeries server

The following sections explore the Domino components in greater detail.

4.2.1 Transactions in Domino

In Domino, a *transaction* is an action that takes place. It requires a response from the Domino server to complete the task, for example, open, read, update, and close. Generally, Domino activities are created from the use of several transaction requests to complete a specific function. Transactions can be used to provide server functions as well as database manipulation. Here, we do not focus so much on the specific function. The focus is on the average response times for a task, which may be comprised of several functions. The goal is to find the average response time of a specific task and of all of the tasks collectively.

4.2.2 Domino base functions

The Domino server has a variety of tasks available to provide functionality. There are some tasks that are common to most implementations. We consider these tasks to be the base functions of Domino. Depending on the function of the task, it is considered either as an interactive task or a background task. For performance tuning purposes, the Domino tasks shown in Table 4-1 are identified as a base Domino function.

Table 4-1 How base tasks are executed within Domino

Tasks	How tasks are executed
One Router thread (Mail Router)	Background task
One Replica Task (Replication)	Background task
AdminP running	Background task
Events and Reporting enabled	Background task
One Updater (Indexing)	Interactive task
C&S enabled (Calendar & Schedule)	Interactive task

Tasks	How tasks are executed
Agent Manager running	Interactive task

Table 4-1 indicates how base tasks are run within Domino. Interactive tasks are directed more toward responding to a Notes Client request. Some tasks are configured with multiple threads from within Domino.

4.2.3 Domino advanced functions

The following Domino tasks are considered to be the advanced tasks in Domino. You can increase performance by isolating resources and priorities when these advanced services are configured on individual servers.

► **Clustering and failover**

Clustering Lotus Domino databases together on one iSeries server provides high availability and failover that can be managed and optimized for performance from a single point of administration. By combining Domino clustering with the ability to configure multiple Domino servers on the same iSeries server, you provide a superior combination from a performance standpoint because clustered database updates do not have to travel across the network.

► **SMTP and HTTP**

By creating SMTP- and HTTP-specific Domino servers as separate servers, you can use the native iSeries performance tools to adjust the priority of these servers to a lower level if desired. With SMTP configured on an individual Domino server, there may be some additional administration. However, if a problem arises with Internet mail, you can recycle the server without affecting the active Lotus Notes users.

By configuring the Lotus Domino HTTP on an individual server, you can minimize the effect of HTTP requests on Lotus Notes users. Also, if an HTTP request begins to loop, you can recycle the HTTP server without affecting the active Lotus Notes users.

4.3 Domino tasks on the iSeries server

Tasks are specified within the notes.ini file load when the Domino server is started. Domino tasks are batch immediate. Therefore, there are no interactive jobs to see. Instead, you must monitor the actual Domino threads. This is accomplished through the standard iSeries Performance Tools. The initial startup of the Server Tasks causes approximately 8 to 10 threads to activate on the OS/400 operating system. Each thread reserves about 4K of memory. When additional users sign on, up to 250 threads become active. This causes the iSeries CPU resource utilization to increase for a short period of time.

One of the most important points to keep in mind when you are working with performance for Domino for iSeries is that you have to deal with two memory managers. By design, the OS/400 operating system has its own memory manager. The Work with System Status (WRKSYSSTS) command provides access to main storage where you can increase pool sizes, the maximum number of active jobs in the pool, and so on. Domino also has a memory management capability that is controlled by parameters in the notes.ini file. If you tune only for iSeries performance, you may see limited or no performance improvement. You must address Domino memory management in conjunction with the iSeries performance tuning. Having the parameters set correctly and working together for both memory management tools provides the best performance results.

Because there is no way to directly see how Domino is performing on the iSeries server, the best way to approach performance is to collect and analyze data from both the Domino and OS/400 environments. By using the Show Transactions command from the Domino console, you can see the actual transactions that are taking place on the Domino server. You can then look at the Domino Log Statistics for that transaction to see the time it took to complete the transaction. You can then analyze the iSeries performance data collected on CPU utilization and memory during the same time frame by using standard iSeries performance reporting techniques.

Although there is no direct comparison, by evaluating the performance factors of both environments, with a little practice, you can develop a better understanding of the relationship between the Domino and OS/400 environments. In Domino, interactive and background tasks run at the same priority. The following tasks can help you tune Domino and the iSeries server:

1. Put Domino in its own system pool.

This makes it much easier to estimate the workload and, therefore, make a prediction on the affect of increased workload.

2. Make a base performance evaluation after the Domino server is started.

As the Domino server goes through ramp up and clients begin to access the server, threads are spawned that cause increased CPU utilization for a short period of time. Information collected during this time frame can be misleading as you evaluate the performance data.

3. Pay attention to the notes.ini file.

The correct balance of the iSeries performance parameters and the Domino performance parameters are essential to overall system performance.

4.4 Application development performance

When you consider the affects of application responsiveness in a Domino environment, there are several factors that can impact performance:

- ▶ The number and frequency of users accessing the application
- ▶ Access to a backend relational database by the application
- ▶ Client type (simple browser, Lotus Notes Client)
- ▶ Client and server operations
- ▶ Domino server architecture
- ▶ Server deployment topology

Although all of the factors mentioned above are important, the factor that has the most impact is the actual design of the applications. By building a complex application, you increase the processing and calculations required by the server, which can compromise performance. Developers must balance the need for clean, simple functionality with the need for performance. There is no single means of maximizing the performance of a Domino application. The desire for application performance does not always leave room for design elegance or maintainability. When designing an application, always start at the high level, where you can make performance considerations from the architectural perspective. The following list contains some application design factors that affect performance. You should carefully consider these points when developing an application with performance in mind.

- ▶ Requirement for agents execution
- ▶ Actions within forms
- ▶ Field validation
- ▶ Using formulas in views
- ▶ Dynamic data versus static data

- Indexing requirements
 - Frequency of indexing
 - Full text indexing

Further information on application development performance for Domino can be found in *Performance Considerations for Domino Applications*, SG24-5602.

4.5 Lotus Domino overall performance on the iSeries server

Today, more customers are buying iSeries advanced servers to use solely as a Domino platform. However, customers frequently want to maximize resources on their current iSeries server. In this environment, you may have a mix of applications with performance criteria that vary over time. This can cause some conflict in performance requirements.

From a system resource utilization viewpoint, it is important to keep these different processing requirements in mind. The Domino installation must be architected from a perspective that keeps the total iSeries server environment in mind. The iSeries hardware and software resources are limited and must be distributed proportionately according to overall organizational objectives.

Placing Lotus Domino on an iSeries server that is already servicing business applications has a significant impact on the performance objectives of that system and requires additional consideration in the future performance management of that iSeries server. Figure 4-1 shows an example of a performance tuning flowchart.

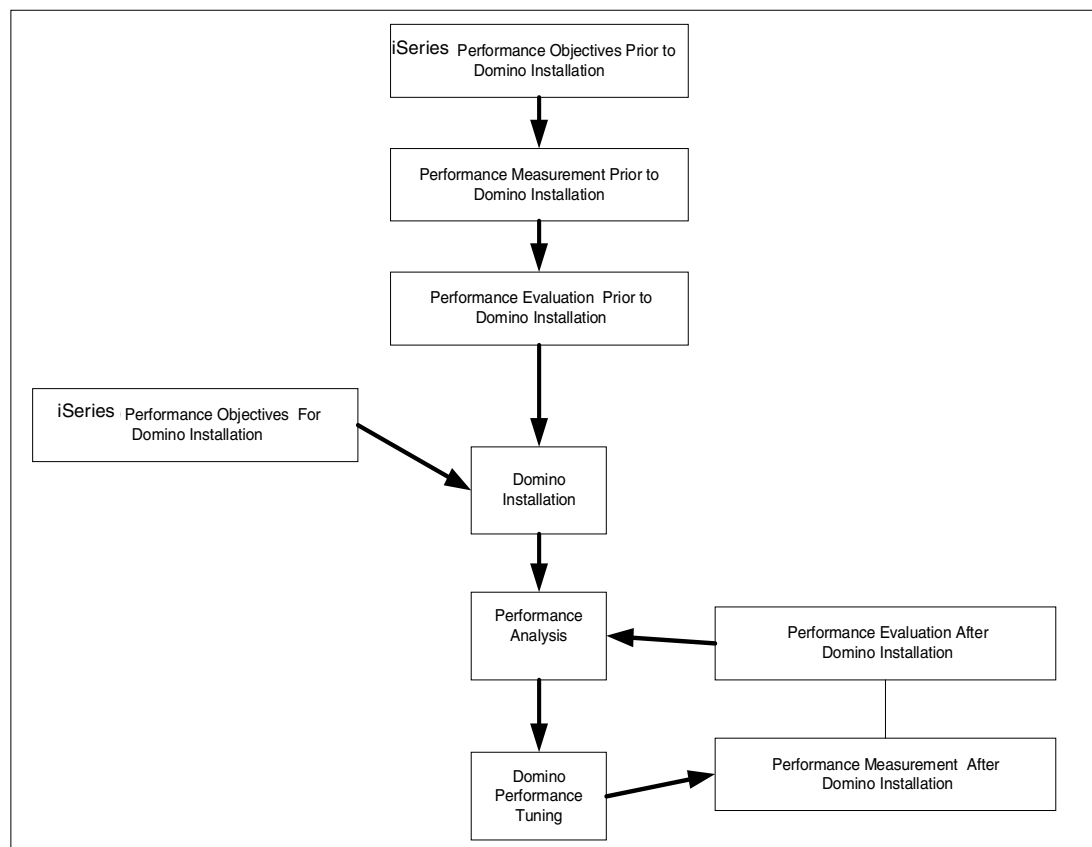


Figure 4-1 Objectives, measurement, and analysis for good performance

A good performance analysis should be completed on the target system before installing Lotus Domino. By properly tuning the iSeries server, you eliminate the possibility of mistaking any pre-existing performance concerns with Domino installation performance allowances.

Once the Domino installation is complete, performance measurement and analysis should continue on a frequent basis for an extended period of time. The conclusion from this analysis may change over time. However, it provides the necessary foundation for making proper performance adjustments.

4.6 The Queuing Multiplier (QM) curve

Domino on the iSeries, as with any application, is affected by many of the same performance factors that are common to the iSeries operating system. It is important to give some considerations to these common performance factors to make sure that you have a firm understanding of the iSeries performance requirements and the affect that they have on the Domino server. System performance may decrease primarily because of the workload. However, it can also be affected by improper system tuning. On the iSeries server, wait times, including waiting for CPU and disk I/O, are associated with queuing for the server. The higher the server utilization is, the greater the wait or queuing time is.

In general, the time it takes for a request to be serviced depends on three primary parameters:

- ▶ The number of waiters in the line ahead of a new request
- ▶ The number of servers responding to requests
- ▶ The service time to complete a request once given to the server, which is a function of the speed of the server and the amount of work to do

The Queuing Multiplier is an important factor when projecting the impact of adding work or additional hardware on current system performance. As the utilization of the a server increases, queuing accounts for a part of the longer work (for request) completion. The Queuing Multiplier is a measure of the affect of queuing.

Table 4-2 shows a simple example of describing the affects of queuing in practice. Let us assume that a transaction completes in one second when CPU utilization is 10%. If the CPU utilization is 50%, it takes two seconds to complete the transaction. If the CPU utilization is 90%, it takes 10 seconds for the transaction to complete.

Table 4-2 How queuing factors affect response time

CPU utilization	Queuing Multiplier	Response time
10	1.11	1 second
50	2	2 seconds
70	3.3	3.3 seconds
90	10	10 seconds

Using a simple example, assume that the CPU is 67% utilized. The mathematical equation says the QM for a single CPU is three. A QM of 3 means, on the average, that there are a total of three requests in the queue (you and two requests for work ahead of you).

The Queuing Multiplier values used in the formulas for disk and CPU service time are shown graphically in Figure 4-2. This shows the utilization at various rates and the significance of the knee in the curve for each type of N-way processor. Remember, after the knee, the service times become less stable and may increase dramatically for small utilization changes.

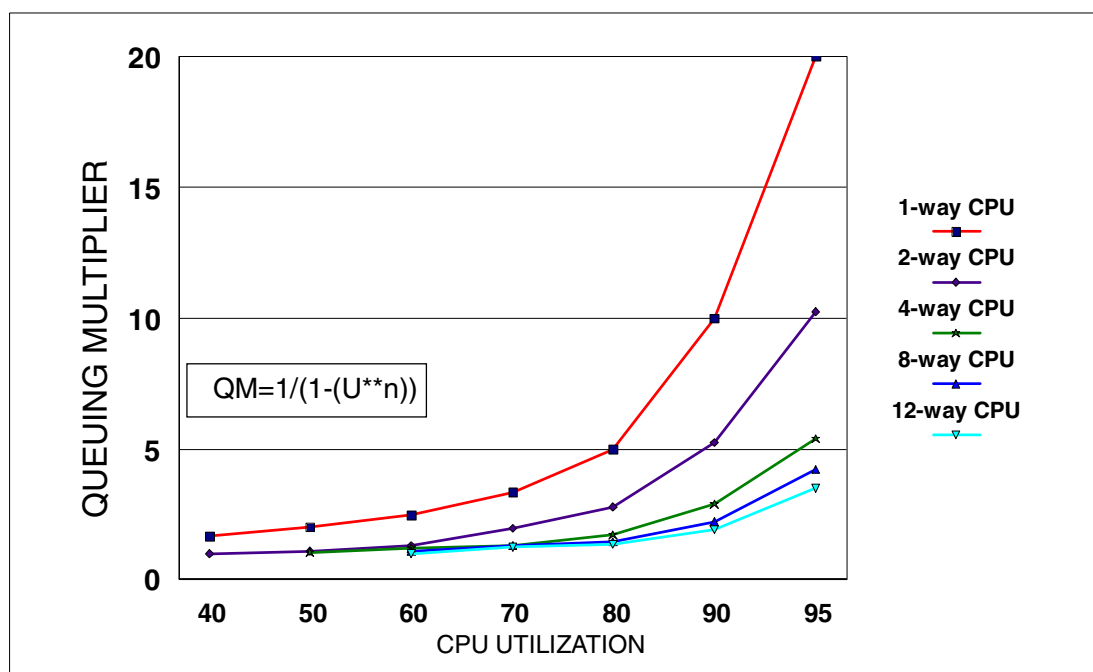


Figure 4-2 Queuing Multiplier versus CPU utilization

Note: You may expect performance problems as the Queuing Multiplier increases above 4. This is a conservative guideline since multi-processor systems can provide acceptable performance with higher Queuing Multiplier values. The CPU queuing multiplier is shown on the Job Summary Report of Performance Tools.

4.7 Dividing the CPU utilization

This section discusses CPU usage for jobs run on the iSeries server. The amount of CPU cycles used are divided among all the jobs in the system. In this example, we assume:

- ▶ Batch jobs use 30% of the CPU.
- ▶ Interactive jobs use 20% of the CPU.
- ▶ Domino jobs use 50% of the CPU.

This is demonstrated in Figure 4-3.

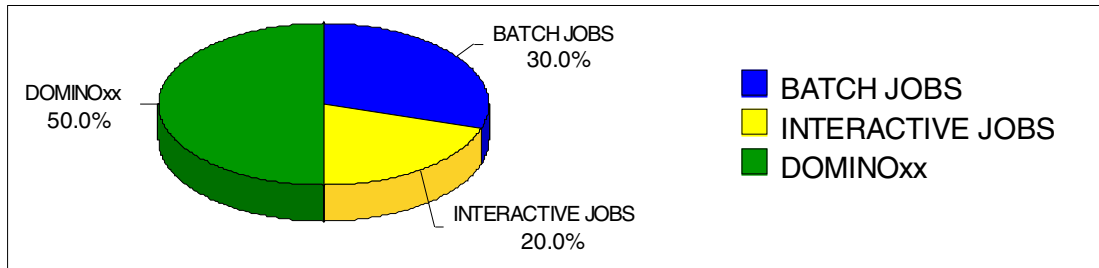


Figure 4-3 Dividing the CPU with all jobs in the system

To introduce more jobs, you need a larger CPU. Therefore, it is necessary to choose the correct iSeries configuration for the number of users and the jobs that the users will perform.

Figure 4-4 shows how 50% of the CPU used by Domino can be shared between tasks:

- ▶ The server consumes 45% of the CPU.
- ▶ The router task consumes 25% of the CPU.
- ▶ The update task consumes 30% of the CPU.

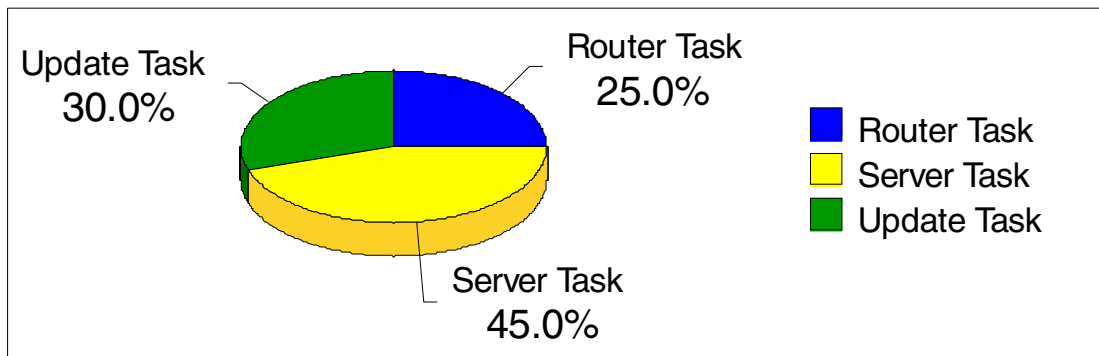


Figure 4-4 Dividing the CPU time used between the Domino tasks

4.8 A brief discussion of threads

This section provides a brief description of:

- ▶ Thread concepts
- ▶ Threads on the iSeries server
- ▶ Domino server threads

You can find additional information by searching for threads on the iSeries Information Center site at: <http://publib.boulder.ibm.com/html/as400/v5r1/ic2924/index.htm>

We discuss threads because any Domino subsystem is actually a multi-threaded client/server application. The term *thread* is shorthand for “thread of control”. A thread is an independent unit of execution within a program. Using threads is a programming technique to run various parts of an application concurrently. For example, one thread may be writing a file while another thread is receiving input from a communication line. A process is the container for the memory and resources of the program. Process resources are accessible to all threads.

Each piece of work on the system is performed in a job and each job has a unique name within the system. All jobs, with the exception of system jobs, run within subsystems. A job enters the subsystem from any of the work entries, such as, a job queue entry, workstation entry, communications entry, autostart job entry, or prestart job entry.

A *job* is a collection of one or more threads. Each job has at least one thread, which is identified as the initial thread. The initial thread is created when the job starts. The job may also have additional threads, identified as secondary threads, depending on the applications that are used by a job. A thread inherits the run attributes from a job initiating the thread. They all start at the same run priority and use a time slice of the same length.

The thread is an independent unit of dispatchable work. Each thread has its own run environment, such as a call stack. However, a thread shares many of the resources that are assigned to the job. The identifier for the thread is unique in the job to which the thread belongs.

Controlling work in the system is performed mainly at the job level. Most commands and application programming interfaces (APIs) operate against the entire job.

Resources are owned at the thread level or the job level. The job serves as an owner for resources that are shared with all threads within the same job. Similarly, some attributes that determine how work is processed are defined at the thread level and some are defined at the job level. Individual attributes and interfaces need to be reviewed to determine the scope of the attribute.

Information about how work is processed is described throughout this redbook. When the information about the work is referred to as a job, the information applies to all threads as a single unit within the job. When the information about the work applies to an individual thread within a job, the information refers to only the thread.

An example of a topic that refers to threads rather than jobs is the information about maximum activity levels. Because each thread is an independent unit of work, the activity level of a storage pool applies to threads rather than jobs. However, the maximum active counts associated with a subsystem and the subsystem work entries apply to jobs. Therefore, a thread is used in information about storage pool activity levels. A job is used in information about subsystem maximum active counts.

4.8.1 Threads on the iSeries server

On the iSeries server, a job represents a process. Each process has at least one thread (a task) in which the program runs its code. The first thread in a process is referred to as the *initial thread*. Some processes are capable of supporting additional threads, which are called *secondary threads*.

Since V4R2, OS/400 supports a *kernel thread* model. Kernel threads are separate tasks associated with a process. The kernel thread model uses a pre-emptive scheduling policy in which the operating system decides which thread is eligible to share the processor. In a kernel thread model, there is a one-to-one mapping between program threads and process threads.

The cost in system resources for activating a new thread is far less than for a job. The job activation and termination is not associated with a thread.

Asynchronous I/O completion ports (IOCP)

You may have already heard of this support in another term, thread pools. Asynchronous I/O completion ports allows Domino to use fewer threads to support users under the SERVER job. This means that instead of having one thread per client user, a small group or pool of threads handles the work for all users. This means less CPU utilization for the Domino server and a reduction in the amount of memory that is required to handle the client users.

We highly recommend that you take advantage of this support if possible because the performance improvements can be substantial:

- ▶ 10 to 30% improvement in CPU utilization
- ▶ 20 to 24% reduction in memory required

You can tune the number of threads in the pool by specifying the following line in your notes.ini file:

```
SERVER_POOL_TASKS=n
```

By default, the number of threads is 40 per port. In our testing, 40 threads were able to handle 3000 clients with high efficiency. If your server handles fewer clients concurrently, you can improve performance by decreasing the number of threads in the pool proportionately.

This support is available in QMU 5.0.1.02 and later releases of Domino for iSeries. It requires OS/400 level V3R3 or higher.

4.8.2 Other Domino task that use threads

Other Domino tasks include such jobs as Router, Replica Agent Manager, and so on. Some jobs are single threaded, and some are multi-threaded based on organizational requirements.

4.9 A brief discussion on run attributes

A job entering the iSeries server receives its run-time attributes from the class description. This provides the ability for certain Domino tasks or the entire Domino server to use different classes and, therefore, allows more flexibility in adjusting run-time attributes of jobs. Some of the run attributes or parameters that are important to work management are:

- ▶ Machine run priority (RUNPTY)
- ▶ Purge (PURGE)
- ▶ Time slice (TIMESLICE)
- ▶ Default maximum wait time (DFTWAIT)
- ▶ Maximum processing time (CPUTIME)
- ▶ Maximum temporary storage (MAXTMPSTG)
- ▶ Maximum threads

Each of these attributes contributes to how and when a thread is processed.

Note: Maximum processing unit time and maximum temporary storage limits can help prevent an erroneous program from impairing system performance. However, a sufficient amount of processing time and temporary storage must be given to allow the job to complete.

4.9.1 Run priority

Run priority is a value that ranges from 0 (highest priority) through 99 (lowest priority). It represents the importance of a job when it competes with other jobs for machine resources. This value represents the relative (not absolute) importance of the job. For example, a job with a run priority of 25 is not twice as important as one with a run priority of 50.

The value is the highest run priority allowed for any thread within the job. Individual threads within the job may have a lower priority. Changing the run priority for the job affects all threads within the job.

Run priority can also be seen as a variable that defines the rank of a thread. A thread having a high priority number is processed later than one having a small priority number. For example, the system threads run on a priority level of 0, where batch jobs run on a priority of 50. By default, all Domino-related threads run on a priority level of 20, on the same priority level with interactive threads. The following sequence of events is a high-level view of how the iSeries work management handles threads:

1. A job enters the system and starts at least one thread.
2. A thread wants to utilize CPU. It is placed on a queue.
3. The CPU signals its readiness to process anything.
4. The gatekeeper on the queue selects the thread with the highest run priority out of the queue and provides it with access to the CPU.
5. The thread uses the CPU until it has finished with the CPU or it hits the end of its time slice and is thrown out of the CPU. See 4.9.4, "Time slice parameter and tuning" on page 66, for more details.

The iSeries server has classes for batch-like jobs and for interactive jobs. By default, the batch jobs run at a lower run priority than the interactive jobs. This mechanism provides the higher priority threads CPU cycles even if the low priority jobs seem to use up all of the available CPU.

Figure 4-5 shows the graphical representation of the CPU utilization between January 2001 and July 2001 for ACME, Inc.

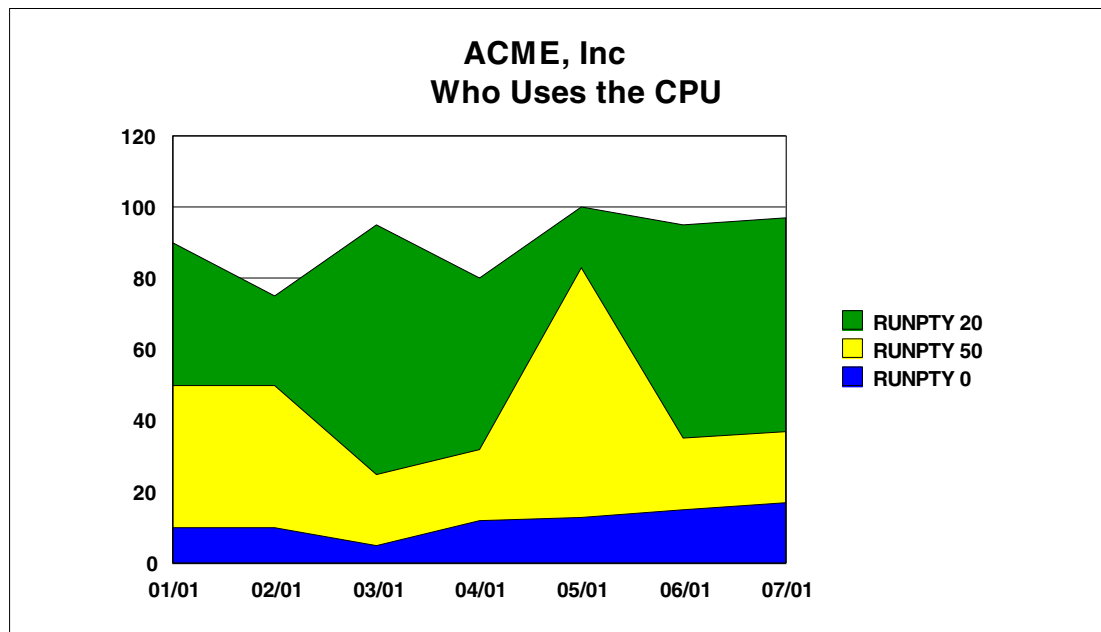


Figure 4-5 How different run priorities use the CPU

The chart in Figure 4-5 shows a summary of the total CPU Utilization per run priority. Pay attention to the period 03/01. Jobs on run priority 50 used 25 percent of the CPU, which left 70 percent for jobs on run priority level 20. This is interpreted as meaning either:

- ▶ There were no jobs running on priority level 50.
- ▶ Jobs on run priority level 20 monopolized the system resources.

During the period 05/01, a different situation occurred. Jobs on run priority level 50 used 70 percent of the CPU cycles, which left only 17 percent for the jobs running on priority level 20.

In both time periods, the overall CPU utilization was about the same. However, different tasks received the CPU time. High priority jobs suffered when low priority jobs flourished. Likewise the high priority jobs flourished when the low priority jobs suffered. The important point is to complete the work, not the amount of CPU utilization.

4.9.2 How a change in run priority affects a job

When you lower a run priority of a thread, you make the thread stay on a queue longer than before. When you raise the run priority of a thread, you *may* decrease the time that the thread waits in a queue, provided that there are now fewer jobs with equal or higher priority competing for resources. Therefore, it gains control of the CPU more often and completes the work faster.

Note: Changing the priority does not necessarily slow down or speed up a job. Only on a busy system do high priority jobs gain access to resources earlier and prevent low priority jobs from running as fast as possible. If a high priority job starts looping, other jobs with lower or equal priority will stop working.

All threads that have the same run priority level are treated equally. If all threads on the system have the same run priority, the CPU is shared according to the first come, first served basis.

4.9.3 Job priority considerations

Normally, the priority for jobs that run in batch environments should be lower than the priority for jobs in interactive environments. The time slice should also be small enough to prevent a looping program from dominating the processor time and activity level in the storage pool.

Note: You can use the QDYNPTYADJ system value to maintain high performance of batch job processing on iSeries servers. For more information about this system value, see 4.10.7, “Work with System Values (WRKSYSVAL) command” on page 79.

4.9.4 Time slice parameter and tuning

Time slice is a performance tuning parameter. The time slice value is specified in the job class discussed in 4.9, “A brief discussion on run attributes” on page 64. The value represents the amount of processing unit time that each thread in a job is allowed to use while processing a transaction. It does not represent the elapsed time of a transaction.

If a thread fails to complete a transaction in the specified time slice, one of the following actions occurs:

- ▶ If no threads of equal or higher priority are on the ineligible queue, the thread is given another time slice, remains in main storage, and continues the transaction.
- ▶ If threads of equal or higher priority are on the ineligible queue, the thread currently running is removed from main storage and placed on the ineligible queue. A thread from the ineligible queue is moved into main storage and processing continues.
- ▶ If the job is interactive and a time slice end pool is specified for the job, the thread is moved to the time slice end pool.

When observing job transitions, we usually advise that you complete a transaction during a single time slice. This reduces the number of times the job enters and leaves main storage. Therefore, the active-to-ineligible transitions should be 0. However, long running transactions should not occupy activity levels for the entire transaction. Establish a time slice value that allows 90% of the transactions in your environment to finish in a single time slice. An example of calculating a time slice value is:

$3 \times \text{the average CPU per transaction}$

4.9.5 How the system manages job and run priorities

The iSeries server attempts to allocate main storage as efficiently as possible. A job may not use the same amount of resources each time it runs. For example, if there are several active jobs on your system, a job spends more time re-establishing the resources needed for running than if a dedicated system environment is used. The system uses the run priorities assigned to different jobs to assist in managing main storage. Therefore, high priority jobs use less system resource than low priority jobs.

4.10 Reviewing iSeries performance using CL commands

To review iSeries server performance interactively, you can use the following CL commands for each area:

► **Processor load**

Use the following commands to determine if there are too many jobs in the system or if some jobs are using a large percentage of processor time:

- Work with Active Jobs (WRKACTJOB)
- Work with System Activity (WRKSYSACT), which is a Performance Tools command

► **Main storage**

Use the Work with System Status (WRKSYSSTS) command to review faulting and the Wait-to-Ineligible transitions.

► **Disk**

Use the Work with Disk Status (WRKDSKSTS) command to determine if there are either too few arms or if the arms are too slow.

► **Communications**

Use the following Performance Tools to find slow lines, errors on the line, or too many users for the line:

- Performance Tools/400 Advisor
- Performance Tools/400 Component Report
- Communications Trace

► **IOPs**

Use the following Performance Tools to determine if any input/output processors (IOPs) are not balanced or if there are not enough IOPs:

- Performance Tools/400 Advisor
- Performance Tools/400 Component Report

► **Applications**

Use the Performance Tools or the Work with Object Locks (WRKOBJLCK) command to investigate locks and mutual exclusion (mutexes).

4.10.1 Work with System Status (WRKSYSSTS) command

You should observe and balance the overall (system wide) performance before you focus on a Domino performance problem. Domino performance is only a relatively small part of the overall performance. If the entire system is functioning poorly, there is no reason to focus on Domino performance.

The Work with System Status display provides a real-time access to data that is also shown in the System Report Storage Pool Utilization section. Use this display to both observe and change the paging fault rates and job transition rates. Pay special attention to the following indicators (in decreasing order of priority):

- ▶ Non-database fault rates in the machine pool
- ▶ Non-database fault rates in all of the other pools
- ▶ Page rates in all of the pools
- ▶ Transition rates in all of the pools

Note: When tuning the system, make sure that the machine pool is adjusted separately from the other pools.

The following examples may help you to understand the performance impact of faulting:

- ▶ The response time of an interactive transaction is affected by any faults that occur during that transaction. Each fault adds from 10 to 30 milliseconds to the end-user response time. For example, if the disk response time is 20 milliseconds and the transaction has five faults per transaction, add about 0.1 seconds to the total response time.
- ▶ Each fault uses a certain amount of the CPU power. As more faults occur, more CPU is consumed for unproductive work.

If the faulting rate of your system is close to the poor end of the faulting guideline tables, approximately 10% to 20% of the CPU is used for faulting. Adding main storage to reduce the faulting rate also lowers the CPU utilization. Therefore, more processing power is available to handle more transactions.

- ▶ With the increasing faulting rate, the amount of disk I/O also increases. If you have only a few arms, these faults can cause the disk utilizations to increase more rapidly than if you have many disk arms. As your disk arm (actuator) utilization increases, the time to process disk I/Os increases and the response times is longer.

While using the Work with System Status display (see Figure 4-6) to analyze a Domino performance problem, you should concentrate on two storage pools. Press the F11 key two times to see the pool names. These storage pools are:

- ▶ ***MACHINE pool:** This is the pool in which the OS/400 jobs and microcode tasks run. Normally, this pool should have the rate of non-DB faults set as close to zero as possible.
- ▶ **Other pools:** This is the pool to which the Domino jobs are routed. The shipped value for this is the *BASE pool. Look at the subsystem descriptions for Domino subsystems to see which storage pool is used by the jobs. Focus on that storage pool.

Then, determine the following information:

- ▶ The faulting rate in the *Machine pool (if the rate is not acceptable, see “The interactive iSeries tuning chart” on page 164)
- ▶ The faulting rate in the storage pool used for Domino jobs
- ▶ The percentage of disk used

Work with System Status									
ACME01									
17/12/2001 06:15:00									
% CPU used	:	26.2	System ASP	:	106.0 G				
Elapsed time	:	00:15:05	% system ASP used . . .	:	76.8542				
Jobs in system	:	776	Total aux stg	:	106.0 G				
% perm addresses	:	.007	Current unprotect used .	:	7345 M				
% temp addresses	:	.015	Maximum unprotect . . .	:	7930 M				
Sys	Pool	Reserved	Max	----	DB-----	--Non-DB---	Act-	Wait-	Act-
Pool	Size K	Size K	Act	Fault	Pages	Fault	Pages	Wait	Inel
1	668800	280884	+++++	.0	.0	.0	.0	11.8	.0
2	1741704	20	123	.0	.0	.0	.0	47.2	.0
3	47184	0	9	.0	.0	.0	.0	.0	.0
4	1260104	0	17	.0	.0	.9	1.9	23.6	.0
5	1000800	316	1775	.0	.0	.0	.0	3139	.0
Bottom									
==> F21=Select assistance level									

Figure 4-6 Work With System Status display

4.10.2 Information about thread state transitions

You can balance your main memory and CPU utilization by allocating the available memory and setting the activity levels in the storage pools. An active thread exists in main storage and processes work requested by the application. A thread in the wait state needs a resource that is not available. An ineligible thread has work to do, but the system is unable to accept more work at that time.

When a thread is waiting for a resource, it may wait in main storage or it may be removed from main storage until the resource is available. The terms *short wait*, *short wait extended*, and *long wait* are used to describe a thread waiting for a system resource.

When observing job transitions, we advise that you allow a transaction to complete during a single time slice. This reduces the number of times the job enters and leaves main storage. Therefore, the active-to-ineligible transitions should be 0. However, long running transactions should not occupy activity levels for the entire transaction. You should establish a time slice value that allows 90% of the transactions in your environment to finish in a single time slice. An example of a time slice value is:

$3 \times \text{the average CPU per transaction}$

Short wait

A thread in *short wait* holds an available activity level while waiting for an activity to occur. A thread can remain in short wait for a maximum of two seconds. When using remote lines, avoid actions that cause short waits because they cause the wait time in main storage to be much longer for a thread than if the thread is waiting for resources at a local workstation.

Short wait extended

A thread is in short wait extended if it is in short wait for the maximum two seconds. After it is in short wait for two seconds and activity has not occurred, the system cancels the short wait, takes the thread out of the activity level, and puts the thread into a long wait. In the performance reports, this thread state transition is called a *short wait extended*.

Long wait

A thread that immediately leaves the activity level is in what is called *long wait*. A specialized form of long wait, called *key/think wait*, occurs outside the activity level when a thread completes a work assignment and returns to request more work. This is the amount of time it usually takes the user to decide what data should be entered and to type the data. When the thread receives a new assignment, it attempts to run again. If no activity level space is available, the thread becomes ineligible.

Other examples of long waits are:

- ▶ Record lock conflicts (when two or more threads attempt to lock the same record of a file)
- ▶ Distributed Data Management data requests
- ▶ Data queue requests
- ▶ Tape read

Ineligible queue and job priority

Sometimes threads gain priority if they are held by locks. Normally, if a thread enters the long wait state by other than a lock conflict, it is placed behind all other threads of equal priority already on the ineligible queue. This is called a *first-in, first-out priority queue*.

However, if a thread becomes ineligible after a short wait extended or a long wait caused by a lock conflict, it is placed in front of threads of equal priority already on the ineligible queue.

The most common reasons for this change to normal queue placement are:

- ▶ The thread enters a long wait as a result of a lock conflict because it was active (referring to objects in main storage) before the conflict occurred. If the wait is short (and many are), you may get the thread back into an activity level before all of the objects in use by the thread are removed from main storage.
- ▶ When the thread is granted the lock, it leaves the wait state. If other threads on the ineligible queue want to use the same object, they must wait until the object is available again. Therefore, you want to use threads holding locks on objects and make them available for other threads to use. To accomplish this, the thread moves ahead of any potential requesters.

Note: Locks can be thread scoped or job scoped. Locks that are job scoped are shared by all of the threads that are in that job.

4.10.3 Work with Active Jobs (WRKACTJOB) command

The WRKACTJOB command shows system performance by measuring aspects such as CPU usage and response time. The threads column shows you the number of threads for each job. Figure 4-7 shows an example of the Work with Active Jobs display.

Work with Active Jobs					ACME01	
CPU %:	97.8	Elapsed time:	00:00:01	Active jobs:	260	
Opt	Subsystem/Job	User	Number	Type	CPU %	Threads
	DOMINO	QSYS	514292	SBS	.0	1
	ADMINP	QNOTES	514325	BCI	.0	1
	AMGR	QNOTES	514321	BCI	.0	1
	AMGR	QNOTES	514324	BCI	.0	3
	CALCONN	QNOTES	514331	BCI	.0	1
	EVENT	QNOTES	514334	BCI	1.1	8
	QNNINSTS	QNOTES	514293	BCH	.0	1
	REPLICA	QNOTES	514307	BCI	.0	1
	REPORT	QNOTES	514335	BCI	.0	1
	ROUTER	QNOTES	514310	BCI	4.2	4
	SCHED	QNOTES	514328	BCI	.0	1
	SERVER	QNOTES	514300	BCI	15.9	257
	SERVER	QNOTES	514343	BCI	45.6	256
	SERVER	QNOTES	514344	BCI	22.9	256
	UPDATE	QNOTES	514315	BCI	.7	1
	UPDATE	QNOTES	514318	BCI	1.3	1
==>						
F21=Display instructions/keys						

Figure 4-7 Work With Active Jobs display

To view the Work with Active Jobs display, type WRKACTJOB on any command line, and press the Enter key. To start (or end) an automatic refresh of information on the display that you are viewing, press the F19 key. The information is updated automatically at time intervals that are specified on the INTERVAL parameter. See *CL Reference*, SC41-5722, for details about the WRKACTJOB command.

While using the WRKACTJOB command, position the cursor on a column and press the F16 key. The items that appear are sorted according to the entries in that column. Use the F11 key to toggle between the different views of the WRKACTJOB display.

4.10.4 Work with System Activity (WRKSYSACT) command

The WRKSYSACT command is a part of the Performance Tools for iSeries (5722-PT1) licensed program and actually provides an enhancement to the Work with Active Jobs display (see Figure 4-8). It is the only tool that shows both jobs and internal OS/400 tasks at the same time on the display. By default, the jobs are sequenced by CPU usage. You can also sequence the display by I/O usage.

Important: WRKSYSACT shows individual measurements for each time interval (snap shots), where WRKACTJOB calculates average values until you press F10. However, if you use the parameter OUTPUT(*FILE) or OUPUT(*BOTH), a database file will be generated with entries for each measurement interval. See 4.11.2, “Using Work with System Activity (WRKSYSACT) with an output file” on page 89.

The WRKSYSACT command is a part of Performance Tools/400 licensed program product. Figure 4-8 shows the panel displayed by this command when you use the parameter OUTPUT(*), the default, or OUTPUT(*BOTH).

Work with System Activity							ACME01			
							11/09/01	15:33:00		
Automatic refresh in seconds							5			
Elapsed time :		00:00:13		Average CPU util . . :		.5				
Number of CPUs . . . :		4		Maximum CPU util . . :		.7				
Overall DB CPU util :		.0		Minimum CPU util . . :		.4				
Type options, press Enter.										
1=Monitor job 5=Work with job										
					CPU	Total	Total	DB		
Opt	Job or Task	User	Number	Thread	Pty	Util	Sync I/O	Async I/O	CPU Util	
	QYPSJSVR	QYPSJSVR	164955	00000004	10	.0	0	0	.0	
	QYPSPFRCOL	QSYS	164938	0000000A	1	.0	0	3	.0	
	QYPSPFRCOL	QSYS	164938	00000009	1	.0	4	0	.0	
	RMMCHDATAC				0	.0	5	0	.0	
	QYPSPFRCOL	QSYS	164938	0000000B	1	.0	0	10	.0	
									Bottom	
F3=Exit			F10=Update list		F11=View 2		F12=Cancel		F19=Automatic refresh	
F24=More keys										

Figure 4-8 Work With System Activity display

What to look for

The Performance Tools WRKSYSACT command provides a list of active jobs that used CPU in the last few seconds. You can determine the specific time by measuring the time interval between uses of the refresh key, PF10. The list is sorted by the amount of CPU seconds used. A high priority job (identified by a low number) may use too much CPU (>50%) for an extended period of time. This may cause poor response time for the entire system.

Note the following points:

- ▶ The *Pty* column shows the run priority of the job.
- ▶ The *CPU Util* column shows the CPU utilization of the job.
- ▶ The *Total Sync I/O* column shows the total amount of synchronous I/O processing that is required by the job. A low amount of synchronous I/O processing is important, because a job must wait for the completion of the synchronous I/O operation before continuing.
- ▶ The *Total Async I/O* column shows the total amount of asynchronous I/O processing that is required by the job. The amount of asynchronous I/O processing is not as important as the amount of synchronous I/O processing, because a job can continue processing immediately after requesting an asynchronous I/O operation. In a way, an asynchronous I/O operation is similar to a batch job. After submitting the job, you do not have to worry about its completion.
- ▶ The *DB CPU Util* column is new since V4R5 and shows the amount of DB2 processing performed by this job. This is important on a iSeries Dedicated Server for Domino, which has a strict limitation of not allowing more than 15% of the total CPU used for DB2 processing. This value allows you to identify jobs doing too much DB2 processing. See 12.3.1, “DB2 database processing defined” on page 351.

- ▶ Entering 1 on the *Opt* column allows you to monitor this job only. Entering 5 in the Opt column, allows you to access the Work With Job display. You can monitor up to 20 jobs and tasks at a single time.

By pressing the F11 key, you can select from three different displays: the summary, synchronous I/O details, and asynchronous I/O details.

The following information about jobs and CPU utilization provides additional tips to improve performance:

- ▶ If a job or small set of jobs is using a large percentage of the CPU, check the job priority (PTY). If the priority of the job that is using too much CPU is a lower number (higher priority) than the jobs with poor performance, you may want to specify a larger RUNPTY. Use option 5 (Work with job) and option 40 (Change job) to specify a larger RUNPTY for the job or jobs that are using too much CPU.
- ▶ If the job that is using too much CPU is an interactive job that is running a batch function (such as compile or test case), the user can submit work to batch or change the priority to 50.
- ▶ If the CPU utilization is high (greater than 80%), but all jobs seem to have an equal amount of CPU usage, the system may have too many active jobs.

4.10.5 Work with Disk Status (WRKDSKSTS) command

Figure 4-9 shows an example of the Work with Disk Status display.

Elapsed time: 00:00:17										
Unit	Type	Size (M)	% Used	I/O Rqs	Request Size (K)	Read Rqs	Write Rqs	Read (K)	Write (K)	% Busy
1	6607	4194	80.8	2.3	4.9	.2	2.1	4.0	5.0	0
2	6607	3145	81.0	1.2	6.8	.4	.8	12.5	4.0	2
3	6607	3145	80.8	.8	17.8	.1	.6	4.0	21.3	0
4	6607	4194	80.8	6.7	6.4	.4	6.2	21.0	5.3	0
5	6713	6442	80.8	3.6	5.0	.2	3.3	12.0	4.4	0
6	6607	3145	80.8	6.5	4.2	.2	6.3	6.0	4.2	0
7	6607	3145	80.8	2.2	4.9	.1	2.0	10.6	4.4	0
8	6607	4194	80.8	18.1	4.5	.3	17.7	15.3	4.3	0
9	6713	6442	80.8	6.7	7.6	1.3	5.3	8.3	7.5	2
10	6607	3145	80.8	4.7	6.0	.0	4.6	116.0	4.6	0
11	6607	3145	81.0	1.3	5.0	.1	1.2	18.0	3.8	2
12	6607	3145	81.0	1.2	6.6	.1	1.1	28.0	4.4	0
13	6607	4194	80.8	.7	5.5	.1	.5	5.3	5.6	2
										More...
Command										
====>										
F3=Exit F5=Refresh F12=Cancel F24=More keys										

Figure 4-9 Work With Disk Status command display

Note: This display may vary up to 20% from actual values. *CL Reference*, SC41-5722, contains a description of the Work with Disk Status (WRKDSKSTS) command and formatting information.

Before you observe the disk status, tune your system. When viewing the Work with Disk Status display, observe the percent of busy data. Each unit (arm) should be less than 50% busy. If each unit is between 50% and 70% busy, you may experience variable response times. If each unit is more than 70% busy, you do not have enough actuators (arms) to provide good performance. The actuator is the device within an auxiliary storage device that moves the read and write heads. If you have a well-tuned system with actuators that exceed 50% busy, you should increase the number of disk actuators.

It is possible to experience inadequate performance even if only one actuator exceeds the 50% busy guideline. This can be caused by the placement of frequently used data on a single storage device. If this occurs on your system, use the Performance Tools for iSeries licensed program disk report to determine which data is causing heavy usage. You can save, delete, or restore some objects to improve performance.

An actuator may exceed the 50% guideline for a short period of time. This condition can be caused by a batch job that is accessing data. If the data is not concentrated on a particular storage device, the high level of use should move from actuator to actuator while the batch job is running. Also, an actuator in an auxiliary storage pool (ASP) may be used heavily. Typically, this is not considered to exceed the guidelines. If you observe this activity, *do not* change the disk configuration.

4.10.6 Observing network performance

While observing the performance, it is important not to forget to verify that the network is not the bottleneck. If you had all the CPU power in the world but a bad network, your users would suffer from poor response times or low transaction throughput.

The PING command

The PING command sends an echo to an IP host to determine whether the host is accessible. The response time shows how long the round trip message takes. This is the first tool to use to identify connectivity problems.

The PING command can be issued on any workstation that supports TCP/IP, for example:

- ▶ Workstation client command line
- ▶ OS/400 command line
- ▶ UNIX client

In Figure 4-10, the PING command sends a data packet of 256 bytes five times in a row. The minimum, maximum, and average response times are all 10 milliseconds.

```
PING RMTSYS(ACME01)
Verifying connection to host system ACME01 at address 1.1.1.1.
PING reply 1 from 1.2.3.4 took 10 ms. 256 bytes. TTL 64.
PING reply 2 from 1.2.3.4 took 10 ms. 256 bytes. TTL 64.
PING reply 3 from 1.2.3.4 took 10 ms. 256 bytes. TTL 64.
PING reply 4 from 1.2.3.4 took 10 ms. 256 bytes. TTL 64.
PING reply 5 from 1.2.3.4 took 10 ms. 256 bytes. TTL 64.
Round-trip (in milliseconds) min/avg/max = 10/10/10
Connection verification statistics: 5 of 5 successful (100 %).
```

Figure 4-10 PING command output example

If you experience response times between a hundred and a thousand milliseconds or even time-outs, use the Trace Route (TRACERT) command to obtain additional information.

The JPING tool

JPING is a GUI tool that is along the same lines as NPING but has additional features over NotesCONNECT (NPING). You can download this tool from the Web at:

<http://www.notes.net/sandbox.nsf?OpenDatabase>

The NPING tool

NPING is a GUI tool to test the various TCP/IP port connections to a particular server. You can download the tool by accessing it from the Web at:

<http://www.notes.net/sandbox.nsf?OpenDatabase>

Trace Route (TRACERT) command

The Trace Route (TRACERT) command is similar to the PING command except that it reports the round trip time by intermediate hosts. It provides information about how each connection between the source system and the target system is performing. An example of the TRACERT output is shown in Figure 4-11.

```
C:\>tracert acme01

Tracing route to acme01.acmeland.acme.com [1.1.1.1]
over a maximum of 30 hops:

  1  <10 ms  <10 ms  <10 ms  acme1342.acmeland.acme.com [1.2.3.4]
  2   20 ms   10 ms  <10 ms  acmebb11.acmeland.acme.com [1.4.5.11]
  3  <10 ms   10 ms   51 ms  acme01.acmeland.acme.com [1.1.1.1]

Trace complete.

C:\>
```

Figure 4-11 TRACERT command output example

Note: The three times reported for each line show that there were three attempts for each connection.

Work with TCP/IP Network Status (NETSTAT) command

Use the Work with TCP/IP Network Status (WRKTCPSTS) command, also known as Network Status (NETSTAT), to obtain information about the status of TCP/IP network routes, interfaces, TCP connections, and UDP ports on your local system. You can also use NETSTAT to end TCP/IP connections and to start or end TCP/IP interfaces.

If Internet Protocol (IP) over SNA (IPS) is enabled, NETSTAT displays information about the IP over SNA interfaces, routes, and connections. You can also use NETSTAT to end IP over SNA connections and to start or end IP over SNA interfaces.

To use this command, the TCP/IP protocol stack or IP over SNA must be active. If neither is active, NETSTAT sends an escape message. Use the display shown in Figure 4-12 to view the status of your TCP/IP and IP over SNA interfaces, routes, and connections.

```

Work with TCP/IP Network Status
System:  ACME01

Select one of the following:

1. Work with TCP/IP interface status
2. Display TCP/IP route information
3. Work with TCP/IP connection status

Selection or command
===>

F3=Exit  F4=Prompt  F9=Retrieve  F12=Cancel

```

Figure 4-12 Work With Network Status selection display

To select an option, type the option number and press Enter. To run a command, type the command and press Enter. For assistance in typing the command, type the command and press F4. For assistance in selecting commands, press F4 without typing anything.

Option 1: Work with TCP/IP Interface Status

Select option 1 to view the Work with TCP/IP Interface Status display (Figure 4-13).

```

Work with TCP/IP Interface Status

Type options, press Enter.
5=Display details  8=Display associated routes  9=Start  10=End
12=Work with configuration status  14=Display multicast groups

Opt  Internet      Network      Line      Interface
     Address      Address      Description Status
x.y.z9.ab2  a.b.ze.192  ACMELINE  Active
x.y.z9.ab5  a.b.ze.192  ACMELINE  Active
x.y.z9.ab5  a.b.ze.192  ACMELINE  Active
x.y.z9.ab6  a.b.ze.192  ACMELINE  Active
x.y.z9.ab7  a.b.ze.192  ACMELINE  Active
x.y.z9.ab8  a.b.ze.192  ACMELINE  Active
x.y.z9.ab3  a.b.ze.192  *VIRTUALIP Inactive
x.y.z33.13  a.b.c33.0   TRNLINE   Active
x0.1.1.1    x0.a.y.0    ACMELINE1 Active
x0.1.1.4    x0.a.y.0    ACMELINE1 Active
x0.1.1.5    x0.a.y.0    ACMELINE1 Active
x0.1.1.6    x0.a.y.0    ACMELINE1 Active

F3=Exit  F4=Prompt  F5=Refresh  F11=Display line information  F12=
F13=Sort by column  F24=More keys

```

Figure 4-13 Work with TCP/IP Interface Status display

Use this display to:

- ▶ View detailed information about a particular interface
- ▶ View a list of the routes associated with a particular interface

- ▶ Start or end a TCP/IP interface
- ▶ Work with configuration status
- ▶ View multicast host groups associated with your interface

Option 2: Display TCP/IP Route Information

Select option 2 to view the routes defined for the system (Figure 4-14). The IP address of the route destination is the ultimate destination reached by using this route. When used in combination with the subnet mask and the type of service values, the route destination identifies a route to a network or system. This route destination IP address can be the address of any of these items:

- ▶ Directly attached network or subnetwork
- ▶ Host system
- ▶ Remote subnetwork
- ▶ Remote network
- ▶ Default route

Display TCP/IP Route Information				
Type options, press Enter.				
5=Display details				
Opt	Route Destination	Subnet Mask	Next Hop	Route Available
	a.b.ccc.d0	255.255.255.0	*DIRECT	*YES
	a.b.cc.dd3	*HOST	*DIRECT	*NO
	a.b.cc.dd2	255.255.255.192	*DIRECT	*YES
	a.b.cc.dd2	255.255.255.192	*DIRECT	*YES
	a.b.cc.dd2	255.255.255.192	*DIRECT	*YES
	a.b.cc.dd2	255.255.255.192	*DIRECT	*YES
	a.b.cc.dd2	255.255.255.192	*DIRECT	*YES
	a.b.cc.dd2	255.255.255.192	*DIRECT	*YES
	aa.bbb.ccc.128	255.255.255.128	*DIRECT	*NO
	aa.bb.c.0	255.255.255.0	*DIRECT	*NO
	aa.b.c.0	255.255.255.0	*DIRECT	*YES
	aa.b.c.0	255.255.255.0	*DIRECT	*YES
	aa.b.c.0	255.255.255.0	*DIRECT	*YES
F3=Exit F5=Refresh F6=Print list F9=Command line				
F11=Display route type F12=Cancel F13=Sort by column F24=				

Figure 4-14 Display TCP/IP Route Information display

The *Subnet Mask* column is the actual value of the subnet mask in dotted-decimal notation.

The *Next Hop* column is the Internet address of the first system on the path from your system to the route destination. The value that appears is either *DIRECT or a dotted-decimal Internet address.

*DIRECT is the next hop value of a route that is automatically created. When an interface is added to this system, a route to the network that the interface attaches to is also created. These routes to interfaces have a next hop value of *DIRECT.

For all other routes, the next hop value is the dotted-decimal Internet address (for example, 9.12.4.6) of the Internet Protocol (IP) router that forwards data to the next system in the path to the route destination.

Because the IP router forwards the data, your system and the route destination do not have to be on the same network. However, the IP router must be on a network or subnet to which your system is directly connected.

The *Route Available* column indicates whether this route is available. One of three possible values appears:

- ▶ ***YES:** The route is available.
- ▶ ***NO:** The route is not available.
- ▶ ***DOD:** The route is used for point-to-point (PPP) dial-on-demand, but is currently not available.

Note: If the status of a route changes from unavailable to available, a period of time elapses before your system recognizes that the route is available. To make your system recognize that the route is available, use the Verify TCP/IP Connection (VFYTCPCNN) command (PING) to verify the connection to the router. If this does not work, notify the system operator of the problem.

Option 3: Work with TCP/IP Connection Status

Select option 3 to view or end a TCP/IP connection between a local system and a remote system (Figure 4-15).

Work with TCP/IP Connection Status					
					System: ACME01
Local internet address : *ALL					
Type options, press Enter.					
4=End 5=Display details					
Opt	Remote Address	Remote Port	Local Port	Idle Time	State
*	*	*	ftp-con >	019:01:37	Listen
*	*	*	telnet	001:48:05	Listen
*	*	*	smtp	042:36:54	Listen
*	*	*	domain	042:37:04	Listen
*	*	*	domain	042:37:04	*UDP
*	*	*	tftp	042:36:55	*UDP
*	*	*	snmp	042:36:57	*UDP
*	*	*	drda	042:37:05	Listen
*	*	*	as-svrmap	001:00:53	Listen
*	*	*	exec	042:36:59	Listen
*	*	*	lpd	042:36:59	Listen
					More...
F5=Refresh F11=Display byte counts F13=Sort by column					
F14=Display port numbers F22=Display entire field F24=More keys					

Figure 4-15 Work with TCP/IP Connection Status Display

The Local Internet address field shows which connections are displayed in the Work with TCP/IP Connection Status list. The following information appears in this field:

- ▶ ***ALL** appears when all connections defined on the local system are displayed. This is the default selection when the connections list is first displayed.
- ▶ An Internet address appears when a subset of the list is displayed. Only connections with a local Internet address equal to this Internet address or with a local Internet address of “*” are shown in the list.

The contents of this field can be changed by using F15 to subset by local address.

The *Remote Address* column shows the Internet address of the remote host. An asterisk (*) is shown if either your iSeries server is waiting for a connection to open or if the list entry is for a UDP socket.

The *Remote Port* column shows either the remote host port number or well-known port name. An asterisk (*) is shown if either your iSeries server is waiting for a connection to open or if the list entry is for a UDP socket.

The *Local Port* column shows your local system port number or well-known port name.

The *Idle Time* column shows the approximate length of time since the last activity on this connection. The time is shown in the format *hhh:mm:ss* (hours, minutes, seconds).

4.10.7 Work with System Values (WRKSYSVAL) command

System values contain specifications that you can use to control or change the overall operation of your system. You can display one system value at a time. A list of the system values (shipped with the system) and their initial contents are available in *OS/400 Work Management for Version 5 Release 1*, SC41-5306.

Use the WRKSYSVAL command to verify the setting the system values described in the following sections.

QTOTJOB system value

The QTOTJOB value controls the total number of jobs for which the auxiliary storage is allocated during IPL. The correct setting of this system value can be obtained by entering the WRKSYSSTS command. Pay attention to the value displayed in the Jobs in system field. The number of jobs in the system should *never be greater* than the value specified for the QTOTJOB system value. Add 15% to the number in the Jobs in system field. Before you set this to the QTOTJOB system value, consider the following words of caution:

- ▶ Clear the output queues regularly. OS/400 reserves auxiliary storage for a job as long as there is at least one spooled output file for that job even though the job is inactive. The more files there are in the output queues, the more jobs you see on the Work with System Status display.
- ▶ If you have a high number of spooled files on the system while using the WRKSYSSTS command and you add 15% more to set the QTOTJOB value, you may significantly increase the time it takes to IPL the system. Performance is also affected when using any system functions that search through the system-wide work control block table (WCBT). These functions include the Work with Active Jobs (WRKACTJOB), Work with Jobs (WRKJOB), and Start Subsystem (STRSBS) commands.
- ▶ Consider using the OS/400 Operational Assistant options to clean the obsolete spooled files such as old program dumps from the system. Do this by entering GO CLEANUP on any command line.

You can suspect a wrong value of the QTOTJOB system value if the system seems to slow down periodically for no apparent reason. A heavy batch job would be one visible reason. The system delay situation normally lasts a couple minutes after which the normal processing continues until the previously created job structures are used up and a new system delay situation arises. The value shipped with the operating system is 30, which normally is not large enough.

Note: A change to this system value takes effect only after the next IPL.

QADLTOTJ system value

The QADLTOTJ system value specifies the additional number of jobs that need storage allocated when the initial number of jobs (the system value QTOTJOB) is reached. Auxiliary storage is allocated whenever the number of jobs in the system exceeds the number for which storage has been allocated.

Note: A change to this system value takes effect immediately.

QACTJOB system value

The QACTJOB value controls the initial number of active jobs for which auxiliary storage is allocated during an IPL. The correct setting of this value can be determined by entering the `WRKACTJOB` command. On the right, top corner of the display is the number of active jobs in the system. Determine the highest amount of the active jobs during a busy day, and add 10% to the number. This number is the correct value for the QACTJOB system value. The number of active jobs should not be greater than this value.

You can suspect a wrong value for the QACTJOB system value if the system seems to fall asleep periodically for no apparent reason. The slowed performance situation normally lasts a couple of minutes after which the normal processing continues until the amount of previously created job structures are used up and a new delay situation arises. The value shipped with the operating system is 20, which normally is not large enough.

Note: A change to this system value takes effect only after the next IPL.

QADLACTJ system value

The QADLACTJ system value specifies the additional number of active jobs that need storage allocated when the initial number of active jobs (the system value QACTJOB) is reached. Auxiliary storage is allocated whenever the number of active jobs exceeds the storage which has already been allocated. The amount of storage allocated for each job is approximately 110 KB.

Note: A change to this system value takes effect immediately.

Important: You must keep QACTJOB, QTOTJOB, QADLACTJ, and QADLTOTJ at reasonable values. If you make QACTJOB and QTOTJOB excessively high, the IPL is slower due to excessive storage allocation. If you make QACTJOB and QTOTJOB too small for your environment and you make QADLTOTJ and QADLACTJ excessively large, run-time performance can be impacted.

QMAXACTLVL system value

The QMAXACTLVL value determines the maximum activity level of the system. This is the number of all of the jobs that can compete at the same time for main storage and processor resources. If a job cannot be processed because no activity levels are available, the job is held until another job reaches a time slice end or a long wait. See Chapter 14 of *OS/400 Work Management for Version 5 Release 1*, SC41-5306, for information about job state transitions.

Important: This system value should be set to *NOMAX and the activity levels should be controlled at the pool level.

Note: A change to this system value takes effect immediately.

QMCHPOOL system value

The QMCHPOOL system value affects the size of the machine storage pool. The machine storage pool contains the highly-shared microcode and operating system programs. You must be careful when changing the size for this storage pool because system performance may be impaired if the storage pool is too small. This value can be changed by the performance adjustment support when the QPFRADJ system value is set to 1, 2, or 3.

Note: A change to this system value takes effect immediately. The value shipped with the operating system is 20000KB.

You can also change the setting of the QMCHPOOL system value by using the Work with System Status display as described previously.

QDYNPTYADJ system value

The QDYNPTYADJ system value controls whether the priority of interactive jobs is dynamically adjusted to maintain high performance of batch job processing on iSeries server model hardware. This value is only effective on systems that are rated for both interactive and non-interactive throughput and have dynamic priority scheduling enabled (see the QDYNPTYSCD system value in the following section).

Note: A change to this system value takes effect at the next IPL.

QDYNPTYSCD system value

The QDYNPTYSCD system value allows you to turn on and turn off the dynamic priority scheduler. The task scheduler uses this value to determine the algorithm for scheduling jobs running on the system.

Note: A change to this system value takes effect at the next IPL.

QPRCMLTTSK system value

The system value QPRCMLTTSK allows you to turn on and turn off the Processor multi-tasking capability. If it is enabled, more than one set of task data will reside in each CPU. Not all iSeries hardware will support Processor multi-tasking. If the system does not support Processor multi-tasking and it is changed to on, it will be turned back off during the next IPL. Some workloads may experience increased performance due to caching implications.

On a partitioned system (LPAR, not Domino partitioned servers), this system value controls the behavior of *all* partitioned systems and is changed from the primary partition only. Also, certain guest operating systems, such as LINUX, do not support QPRCMLTTSK set to 1.

Note: A change to this system value takes effect at the next IPL.

4.10.8 Display System Log (DSPLOG) command

The Display Log (DSPLOG) command shows the system history log (QHST). The history log contains information about the operation of the system and system status. The system history log contains the messages sent to the log, the date and time the message was sent, and the name of the job that sent it.

4.10.9 Start System Service Tools (STRSST) command

System Service Tools rarely provide performance-related information. However, they may provide you with additional information. For example, a large amount of errors on a communications line tends to cause error-recovery procedures, which may cause performance problems.

From the SST screen shown in Figure 4-16, select option 1 to start a Service Tool.

System Service Tools (SST)

Select one of the following:

1. Start a service tool
2. Work with active service tools
3. Work with disk units
4. Work with diskette data recovery

Figure 4-16 Selecting to start a service tool

The Start a Service Tool display appears as shown in Figure 4-17.

Start a Service Tool

Warning: Incorrect use of this service tool can cause damage to data in this system. Contact your service representative for assistance.

Select one of the following:

1. Product activity log
2. Trace Licensed Internal Code
3. Work with communications trace
4. Display/Alter/Dump
5. Licensed Internal Code log
6. Main storage dump manager
7. Hardware service manager

Figure 4-17 Start a Service Tool

Select option 1 to use a product activity log. You can see such activities as:

► **Analyze log**

Select option 1 (analyze log) to display or print a summary of product activity entries. This summary is useful for analyzing intermittent and multiple error conditions.

► **Display or print by log ID**

Select option 2 to display or print data from the product activity log by log identifier. The log ID is a unique identifier that ties together all data related to a single error condition.

► **Change log sizes**

Select option 3 to verify or change the amount of storage on a disk unit used for product activity log data.

- ▶ **Work with removable media lifetime statistics**

Select option 4 to display, print, or delete the statistical data logged for the lifetime use of a removable media. Lifetime is the total length of time one of these media allows for information to be read from or written to it.

- ▶ **Display or print removable media session statistics**

Select option 5 to display or print the statistical data logged for a session of a removable media. Session is the length of time one of these media is in position to be read from or written to (read/write heads loaded).

- ▶ **Reference code description**

Select this option to display or print the description of a reference code.

4.11 Collecting performance data

Through V4R5, the Start Performance Monitor STRPFRMON command was available with the OS/400 operating system to allow you to collect performance data. At V5R1, Collection Services replaces Performance Monitor. STRPFRMON is no longer available at V5R1. Collection Services collects performance data for all jobs and tasks running on the system. It allows you to continuously collect performance data with minimal overhead and control what type or types of data are collected.

Collection Services uses one or two jobs to collect data. QYPSPFRCOL is the primary job of Collection Services. It collects data into a management collection object (*MGTCOL). The second job, CRTPFRTDA, is only active if you have chosen the create the database files while the collection is active. Both of these jobs will run under the QSYSWRK subsystem.

For more information on Collection Services, visit the Web site at:

<http://publib.boulder.ibm.com/html/as400/v5r1/ic2924/index.htm>

4.11.1 Starting Collection Services

Collection Services can be started by using any of the following methods:

- ▶ Operations Navigator
- ▶ Performance Tools menu options
- ▶ Management Central APIs

Operations Navigator

In Operations Navigator, Collection Services can be started in several different ways. The information described here only shows one way in which it can be started.

As shown in Figure 4-18, you can right-click the system name, choose **Collections Services** and then **Start Collecting** to begin the steps necessary to start Collection Services.

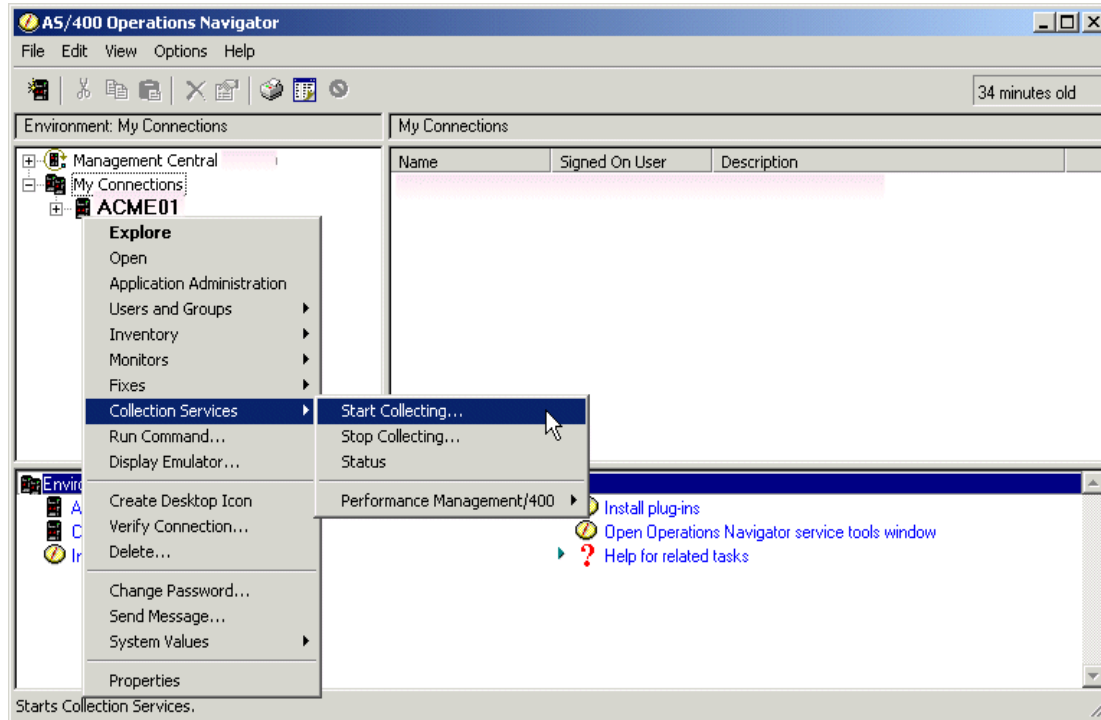


Figure 4-18 Starting Collection Services

For the purposes of collecting sample performance data for Domino performance analysis, Figure 4-19 and Figure 4-20 are set to their default parameters.

Figure 4-19 shows the available parameters that control how the data is collection. From this window, you can specify or change the interval at which to collect the data, when to cycle the collection, and how long to retain the data among other options.

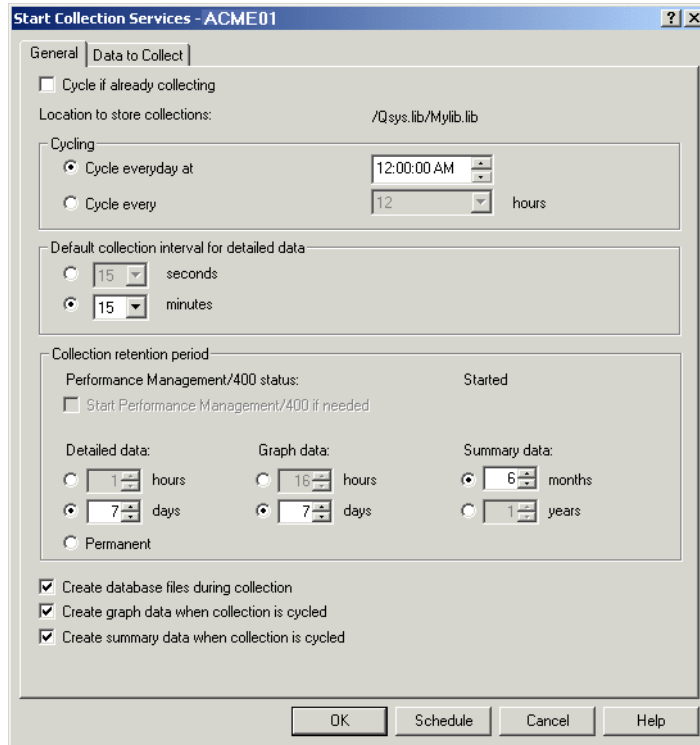


Figure 4-19 General tab from starting Collection Services

Figure 4-20 describes the data that is going to be collected by Collection Services. It is recommended that you leave this parameter set. Changing the parameter to customer allows you to collect different varieties of data which is fine. But eliminating certain points of collection may provide you with insufficient data to do performance analysis.

By clicking **OK**, Collection Services is submitted to QSYSWRK and data is collected. You also have the option to schedule a collection from these screens by choosing the **Schedule** button.

To end the collection, choose **Stop Collecting** from the menu shown in Figure 4-18.

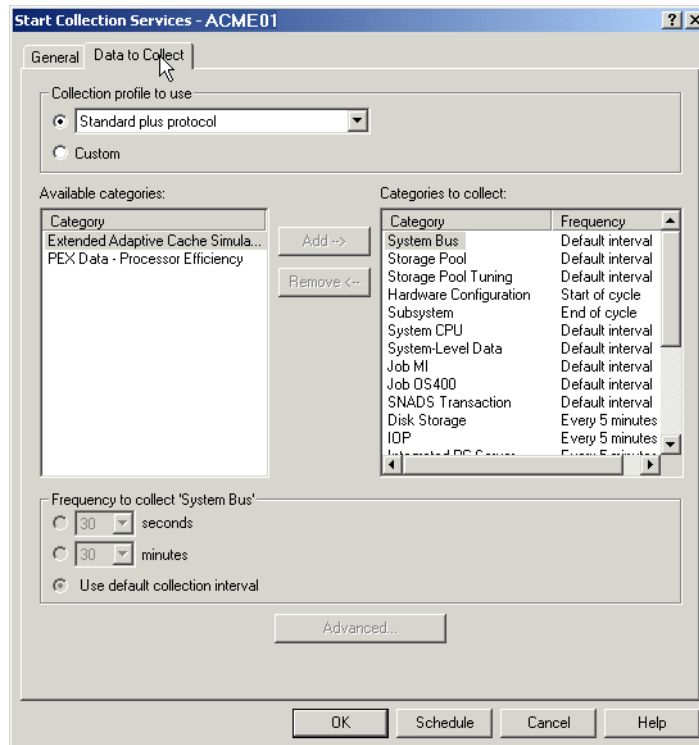


Figure 4-20 Data to Collect tab from starting Collection Services

Performance Tools menu options

By entering the GO PERFORM command on an iSeries command line, you see the IBM Performance Tools for iSeries display (Figure 4-21). The Performance Tools menu is only available if Performance Tools for AS/400 Licensed Program Product (5769-PT1) is installed on the system.


```

PERFORM                      IBM Performance Tools for iSeries
                                System:  ACME01

Select one of the following:

    1. Select type of status
    2. Collect performance data
    3. Print performance report
    4. Capacity planning/modeling
    5. Performance utilities
    6. Configure and manage tools
    7. Display performance data
    8. System activity
    9. Performance graphics
   10. Advisor

    70. Related commands

Selection or command
===>

F3=Exit  F4=Prompt  F9=Retrieve  F12=Cancel  F13=Information Assistant
F16=System main menu
(C) COPYRIGHT IBM CORP. 1981, 2001.

```

Figure 4-21 Performance Tools main menu

To start Collection Services, select option 2 from the menu.

```

                                Collect Performance Data
                                ASHLH
                                10/15/01 10:47:25

Collection Services status:
Status . . . . . : Started
Collection object . . . . . : Q288000001
Library . . . . . : MYLIB
Started . . . . . : 10/15/01 00:00:01
Default collection interval . . : 00:00:15
Retention period . . . . . : 30 days 00 hours
Cycle time . . . . . : 00:00:00
Cycle interval . . . . . : 24
Collection profile . . . . . : *STANDARDP

Select one of the following:

    1. Start collecting data
    2. Stop collecting data

Selection or command
===>

F3=Exit  F4=Prompt  F5=Refresh  F9=Retrieve  F12=Cancel

```

Figure 4-22 Collection Services status display

Figure 4-22 provides the current status of Collection Services. Here you see:

- Status of the collection (started or ended)
- Name of the management collection object
- Library that contains the management collection object
- Interval in which the data is being collected
- How long the management collection will reside on the system
- Time the collection started
- Collection cycle interval
- Type of data being collected

To continue starting a collection, choose option 1.

The Start Collecting Data panel (Figure 4-23) provides the list of parameter necessary to collect performance data. To keep the data centralized, change the library parameter to a library of your choice. All other parameters can be left alone.

Start Collecting Data

Type choices, press Enter.

Library	MYLIB	Name
Collection interval (minutes) . . .	15	0.25, 0.5, 1, 5, 15, 30, 60
Retention period:		
Days	30	*PERM, 0-30
Hours	0	0-23
Cycling:		
Time to synchronize cycle	00:00:00	HH:MM:SS
Frequency to cycle collections . .	24	1-24
Create database files	*YES	*YES, *NO
Collection profile	*STANDARDP	*MINIMUM, *STANDARD, *STANDARDP, *ENHCPCPLN

F3=Exit F12=Cancel

Figure 4-23 Parameter menu for starting Collection Services

When you press Enter at this screen, a new collection is started.

To end the collection, you go into the menu options as shown in the previous steps but choose option 2 (Stop collecting data). You can see this option in Figure 4-22.

Management Central APIs

If neither Operations Navigator nor Performance Tools are available and PM/400 (Performance Management) is not actively collecting data, there are Management Central APIs that provide the functionality to start and end Collections Services.

Starting Collection Services

The Start Collector (QYPSSTRC QypsStartCollector) API starts the collection. If a collection is not currently active, the collection will use the defaults to start collecting data. If there is a collection active it will use the settings from the currently active collection and start a new collection using the attributes from the current collection if no new ones have been specified.

Use the following command to start Collection Services:

```
CALL PGM(QYPSSSTRC) PARM('*PFR      ' '*STANDARDP' X'00000000')
```

Note: There must be six additional spaces after *PFR and before the ending quote on the first parameter.

Ending Collection Services

The End Collector (QYPSENDC QypsEndCollector) API ends an active collection. Use the following command to end Collection Services:

```
CALL PGM(QYPSENDC) PARM('*PFR      ' X'00000000')
```

Note: There must be six additional spaces after *PFR and before the ending quote on the first parameter.

Changing System Collector Attributes

Change System Collector Attributes (QYPSCSCA, QypsChgSysCollectorAttributes) changes system collector attributes. This API provides the default values for Collection Services.

To learn more about these APIs and other Management Central APIs, go to the iSeries Information Center and search for *Management Central APIs*. The Information Center is located on the Web at:

<http://publib.boulder.ibm.com/pubs/html/as400/v5r1/ic2924/index.htm>

4.11.2 Using Work with System Activity (WRKSYSACT) with an output file

The WRKSYSACT command allows you to interactively work with the jobs and tasks currently running in the system. At a first glance, the information shown by this command looks very similar to the Work with Active Jobs (WRKACTJOB) panel. However, the WRKSYSACT has several advantages over WRKACTJOB:

- ▶ It uses less system resources.
- ▶ Besides user jobs, it also shows system tasks.
- ▶ The resource utilization shown is not cumulative and shows as snapshot of each measurement interval. When used with Automatic Refresh (F19) function, it shows live information for the very second, you are watching it.

However, WRKSYSACT is only available, when the OS/400 Performance Tools (5722-PT1) are installed on your system.

Besides having the ability to view this data on the display station, the user can also direct the data to be stored in a database file for future use. To direct the data to a database file, specify either of the following commands:

```
OUTPUT(*FILE)  
OUTPUT(*BOTH)
```

This allows you to produce a report as described in 4.13.4, “Print Activity Report (PRTACTRPT) command” on page 115.

4.12 Management Central performance monitor

Through Management Central, you can manage and monitor your Domino server performance in an easy and understandable way.

System and Job monitors gather and present real-time performance data for your Domino servers. You can use monitors to see your server's performance as it happens at a single system or across multiple systems and groups of systems.

In contrast, you can use Collection Services to collect performance data for later analysis. Collection Services allows you to analyze multiple sets of true performance data for a longer period of your system performance history. Using the Graph History window, you can see a graphical view of the metrics that you collected for an extended period of time. You can use the Graph History function as long as you collect data with Collection Services; you do not need to have a system monitor running.

For information about Collection Services, see 4.11, "Collecting performance data" on page 83.

This section does not cover all the functions in Management Central performance monitors. For more detailed and complete information, we recommend that you read *Managing OS/400 with Operations Navigator V5R1*, SG24-6226.

4.12.1 Monitoring real-time system performance

The system monitor graphs present real-time system performance data in a graphical interface that you can directly manipulate to gather different or more detailed data. Monitors allow you to collect performance data simultaneously for a wide variety of system metrics, for any system or system group, and for any length of time.

To monitor real-time system performance, you start with creating a system monitor.

Creating a new system monitor

You can define a new job monitor by using one of the following methods:

- ▶ Expand **Management Central**. Select **Monitors**. Then right-click **System** and select **New Monitor**.
- ▶ Right-click a system under My Connections. Then select **Monitors->System**.
- ▶ Expand **Endpoint Systems** under Management Central and select a system. Then select **Monitors->System**.
- ▶ Expand **System Groups** under Management Central and select a system group. Then select **Monitors->System**.

Each of these methods activates the New Monitor window shown in Figure 4-24.

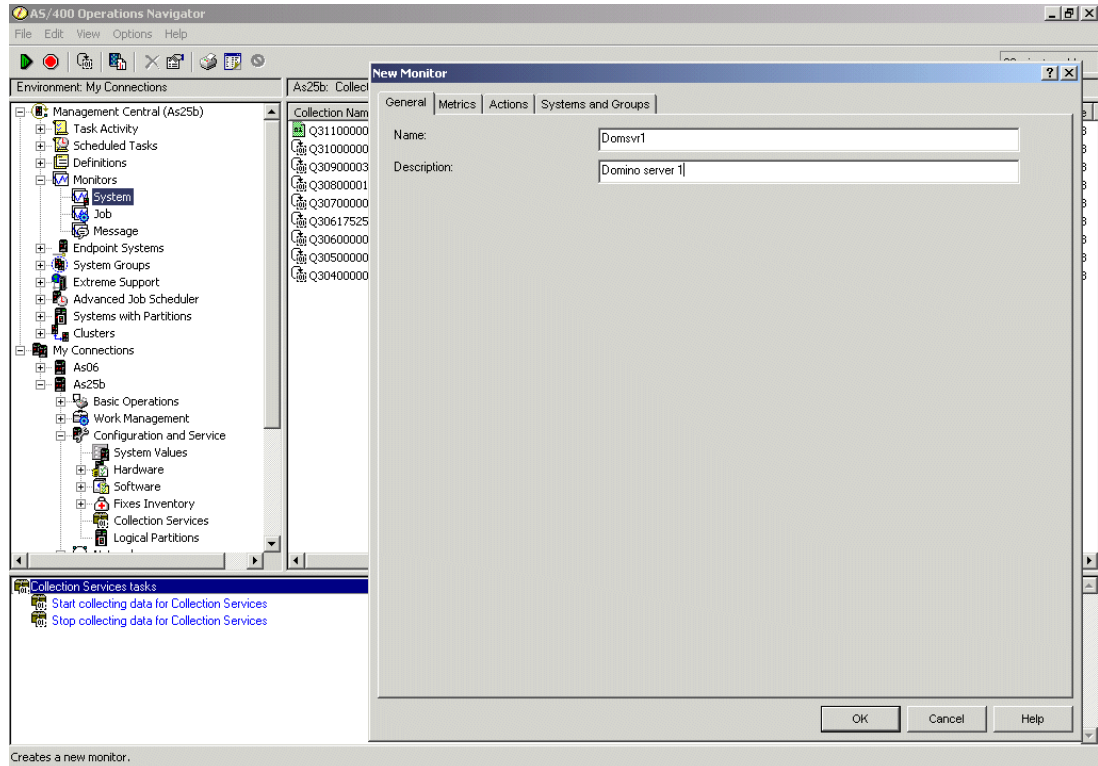


Figure 4-24 System monitor

General

On the General page, you enter the name of the monitor and a brief description of the monitor.

Metrics

The Metrics page for New Monitor or Monitor Properties allows you to select the metrics that you want to monitor. You can view and change information about the collection interval, the maximum graphing value, and the display time for each metric. You can define one or two thresholds for each metric. The thresholds are independent of each other and provide a way to monitor for different conditions in one monitor. For example, you may monitor for a less severe condition and send a command to page the system operator. Or, you may monitor for a more severe condition and send a command page to start ending certain jobs. A threshold consists of a trigger value and, optionally, a reset value. You can specify an OS/400 command to be run when the threshold is triggered or reset.

Before V4R5, you could choose from the following CPU Utilization metrics:

- ▶ **CPU Utilization (Average):** This metric shows you the percentage of available processing unit time that is being consumed by all jobs on your iSeries server. This metric includes all work done on your system, both interactive and non-interactive.
- ▶ **CPU Utilization (Interactive Jobs):** Formerly known as CPU Utilization (Interactive), this metric shows you the percentage of available processing unit time that is being consumed on the system for all jobs of the type I (interactive). It also helps you manage your interactive users' work compared to total CPU utilization capacity. The resulting data is relative to achieving good interactive response time and the amount of CPU left for non-interactive jobs.

Type I jobs include:

- Twinaxial data link control (TDLC)
- 5250 remote workstation
- 3270 remote workstation
- SNA pass-through
- 5250 telnet

- ▶ **CPU Utilization Basic (Average):** This metric shows the percentage of available processing unit time that is being consumed by all jobs on the system. Unlike the CPU Utilization (Average) metric mentioned above, this metric does not track detailed data.

In addition to those metrics available before V4R5, the set of CPU Utilization metrics now includes three *new* metrics. These new CPU Utilization metrics include:

- ▶ **CPU Utilization (Interactive Feature):** This metric is designed to help you monitor and manage your iSeries server's interactive use. It determines whether a particular job is doing interactive work and measures the system's overall interactive workload. This new metric was designed especially with the new iSeries Interactive Feature in mind, but can be used with other iSeries models where an interactive capability is of interest, such as iSeries servers. It complements the existing Management Central metrics such as CPU Utilization (Interactive Jobs) and CPU Utilization (Average). The new metric shows you when your system is approaching its interactive limits:
 - Signed-on 5250 workstation jobs
 - Autostart jobs, prestart jobs, or jobs submitted to a batch job queue that run I/O operations on a 5250 workstation
- ▶ **CPU Utilization (Database Capability):** This metric is intended to help you monitor your iSeries server's database use. Using this metric, you can see how much of your system CPU is consumed by database activities and which jobs contribute the most to this use. In addition, you can find detailed data for each job, including the number of milliseconds of CPU used by that job in database processing during the particular sample interval being graphed.
- ▶ **CPU Utilization (Secondary Workloads):** This metric is designed for use on iSeries dedicated servers. It measures how much CPU is being used on the system for work other than the primary workload for which the system is designed, which can include database activity. For example, this metric can be used on the iSeries Dedicated Server for Domino to see how much non-Domino work is being done on the system. The amount of CPU used by secondary workloads is currently reported only on Dedicated Servers for Domino systems running V4R5 or later and is a portion of the total CPU utilization capacity. On other systems and servers, the value is ignored and appears as 0%. This metric does not track detailed data.

All three new metrics have default threshold values set, that are good starting values.

Note: You have to enable these metrics to enable them for use.

- ▶ **Interactive Response Time (Average):** This includes the average response time for interactive (5250) jobs on the system. Second level information shows the jobs having the highest average response time.
- ▶ **Interactive Response Time (Maximum):** This includes the highest response time for interactive (5250) jobs on the system while the monitor is active. Second level information shows the jobs having the highest response time.

- ▶ **Transaction Rate (Average):** This includes the average number of transactions per second completed by all jobs active on the system. Second level information shows the jobs having the highest rate.
- ▶ **Transaction Rate (Interactive):** This includes the average number of transactions per second completed by 5250 jobs active on the system. Second level information shows the jobs having the highest rate.
- ▶ **Batch Logical Database I/O:** The average number of logical database input/output (I/O) operations currently performed by all non-5250 (batch) jobs on the system. Second level information shows the jobs performing the highest I/Os.
- ▶ **Disk Arm Utilization (Average):** The average percentage of disk arm busy doing I/O operations for all disks on the system. Second level information shows information for each disk arm.
- ▶ **Disk Arm Utilization (Maximum):** The maximum percentage of disk arm busy doing I/O operations for all disks on the system. Second level information shows information for each disk arm.
- ▶ **Disk Storage (Average):** The average percentage of disk arm storage that is full on your system during the time you collect the data. Second level information shows information for each disk arm.
- ▶ **Disk Storage (Maximum):** The highest percentage of disk arm storage that is full on your system during the time you collect the data. Second level information shows information for each disk arm.
- ▶ **Disk IOP Utilization (Average):** The average percent busy the disk input/output processors (IOPs) are on your system during the time you collect the data. Second level information shows information for each IOP.
- ▶ **Disk IOP Utilization (Maximum):** The maximum percent busy the disk input/output processors (IOPs) are on your system during the time you collect the data. Second level information shows information for each IOP.
- ▶ **Communication IOP Utilization (Average):** The average percent busy the communication (LAN, WAN, ...) IOPs are on your system during the time you collect the data. Second level information shows information for each IOP.
- ▶ **Communication IOP Utilization (Maximum):** The maximum percent busy the communication (LAN, WAN, ...) IOPs are on your system during the time you collect the data. Second level information shows information for each IOP.
- ▶ **Machine Pool Faults (Average):** The average number of faults per second occurring in the machine pool of the system during the time you collect the data. Only Licensed Internal Code runs in the Machine pool.
- ▶ **User Pool Faults (Average):** The average number of faults per second occurring in all of the user pools on the system during the time you collect the data. Second level information shows information for each pool.
- ▶ **User Pool Faults (Maximum):** The maximum number of faults per second occurring in all of the user pools on the system during the time you collect the data. Second level information shows information for each pool.
- ▶ **Communication Line Utilization (Average):** The average percentage of line utilization for all non-LAN lines active during the time you collect the data. Line utilization is an approximation of the actual amount of data transmitted compared to the theoretical maximum line speed configured on the line description object. Second level information shows information for each non-LAN line. A non-LAN line is supports binary synchronous, asynchronous, IDLC, X.25, LAPD, or SDLC protocols.

- ▶ **Communication Line Utilization (Maximum):** The maximum percentage of line utilization for all non-LAN lines active during the time you collect the data. Second level information shows information for each non-LAN line.
- ▶ **LAN Utilization (Average):** The average percentage of line utilization for all LAN (Token-Ring and Ethernet) lines active during the time you collect the data. Line utilization is an approximation of the actual amount of data transmitted compared to the theoretical maximum line speed configured on the line description object. Second level information shows information for each LAN line.
- ▶ **LAN Utilization (Maximum):** The maximum percentage of line utilization for all LAN lines active during the time you collect the data. Second level information shows information for each LAN line.

Figure 4-25 shows an example of the metrics that can be added for monitoring Domino server performance. For examples of combining metrics to target specific types of performance information, see 4.12.4, “Combining Management Central’s CPU Utilization metrics” on page 104.

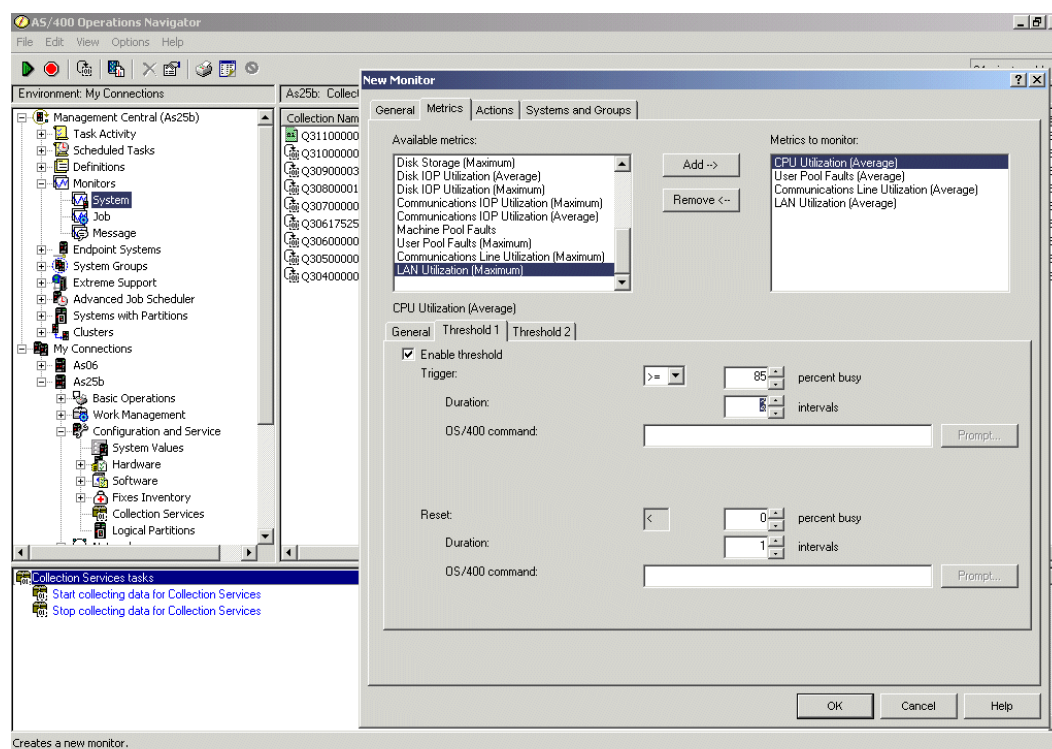


Figure 4-25 System monitor metrics

Actions

The Actions page for New Monitor or Monitor Properties allows you to specify the actions to occur when a threshold is triggered and when a threshold is reset.

- ▶ **Log event:** Adds an entry to the event log on the central system when the threshold is triggered or reset. The entry includes the date and time the event occurred, the endpoint system being monitored, the metric being collected, and the monitor that logged the event.
- ▶ **Open event log:** Displays the event log when a trigger or reset event occurs. This automatically brings up the Event Log window if it is not currently displayed on your PC.
- ▶ **Open monitor:** Displays a list of systems that are being monitored for the specified metrics and a list of the values for the specified metrics as they are collected for each

system when a trigger or reset event occurs. This automatically brings up the Monitor window if it is not currently displayed on the PC.

- **Sound alarm:** Sounds an alarm on the PC when the threshold for the monitor is triggered.

Figure 4-26 shows an example of System monitor Action page.

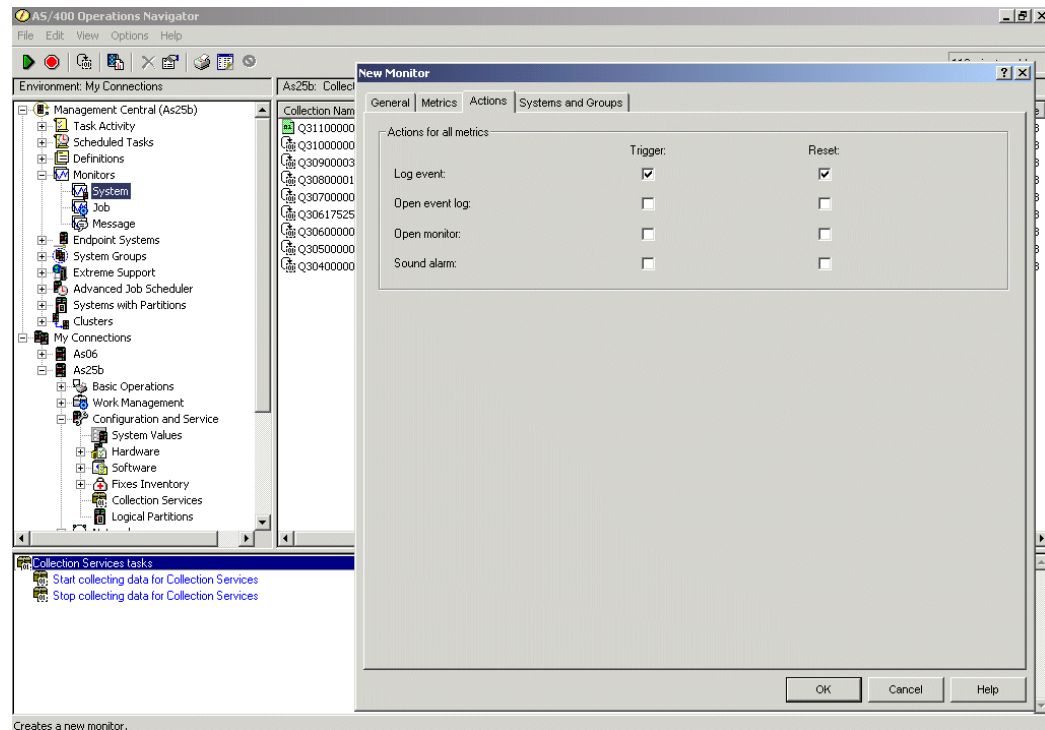


Figure 4-26 System monitor: Action page

Systems and Groups

The Systems and Groups page for New Monitor or Monitor Properties shows you a list of endpoint systems or system groups. The list can include systems from which the monitor is currently collecting data. The list can also include systems on which the monitor is not currently running. You can also add or remove endpoint systems or system groups from this page. Click **Browse** to add systems or groups to your list. Click **Remove** to remove selected systems or groups from the Selected systems and groups list.

Figure 4-27 shows an example of the systems and groups panel.

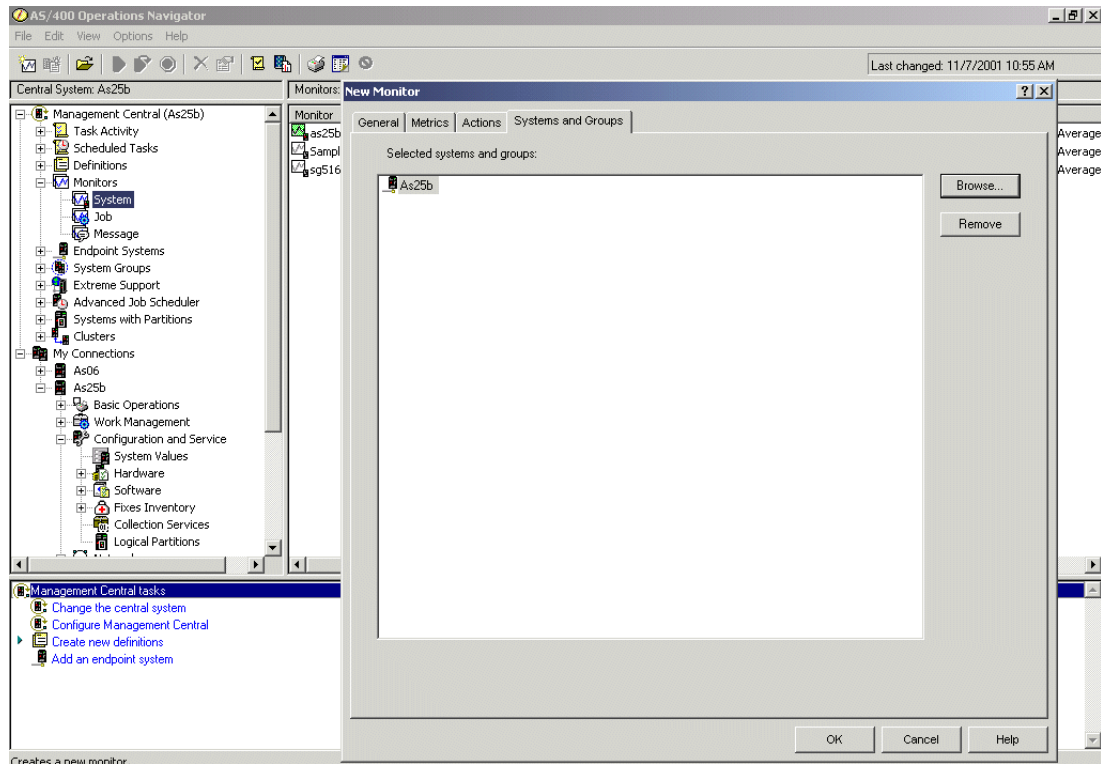


Figure 4-27 System monitor: System and Groups

Modifying an existing system monitor

To modify an existing system monitor, follow these steps:

1. Expand **Management Central** and **Monitors**.
2. Double-click **System**.
3. Select the monitor you want to modify and then select **Properties**.

The Monitor Profile can be changed easily through the Properties option of the Monitor, even while the Monitor is active.

Figure 4-28 shows an example of the window where you modify a system monitor.

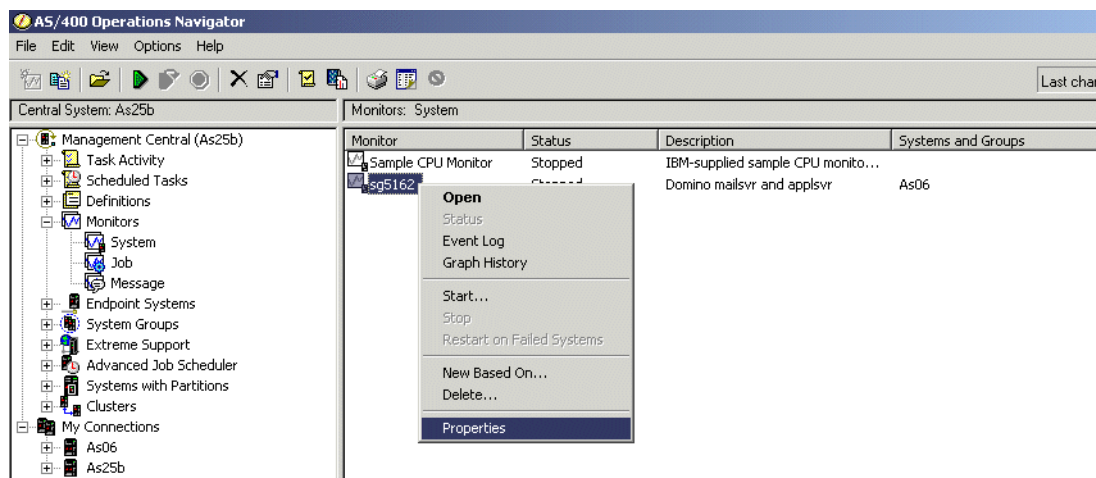


Figure 4-28 Modifying a system monitor

Starting the Real-time monitor

To start the Real-time monitor, double-click the selected monitor, or select the monitor and double-click the green arrow icon (start).

Once you start a monitor, you are free to perform other tasks on your server, in Operations Navigator, or on your PC. In fact, you could turn your PC off! It continues to monitor your systems and perform any threshold commands or actions you specified. Your monitor will run until you decide to stop it.

If the monitor window is closed, the monitor is not stopped. If you want to end the monitor, go to the Operations Navigator session, and end the monitor.

When a Real-time Monitor is started, new threads are associated with the monitor in the Collection Services job QYPSPFCOL and Management Central job QYPSSRV, and in QYPSJSVR jobs in QSYSWRK subsystem.

For more information about the Collection Services job, see 4.11.1, “Starting Collection Services” on page 83.

Time zone considerations: If your systems that are controlled by Management Central are located in different time zones, you must:

- ▶ Ensure that your system values controlling system date (QDATE) and time (QTIME) are set correctly.
- ▶ Be sure that the system value for Coordinated Universal Time Offset (QUTCOFFSET), which reflects the time difference from GMT (Greenwich Mean Time), is set correctly.
- ▶ Consider the impact of changes resulting from transition to and from daylight saving time.

Real-time performance data

The premier feature of monitoring is the graphical view. It shows information based on the metrics selection you make, such as average CPU utilization and average interactive response time throughout the network.

Once the first-level information, known as a *metric*, is graphed, the details about the specific sample or data point can be viewed using the drill-down capabilities. There can be anywhere from 1 to 3 levels of additional data for some of the data points. This drill-down capability gives administrators performance information from across the network.

Figure 4-29 shows an example of the real-time monitoring graphical view.

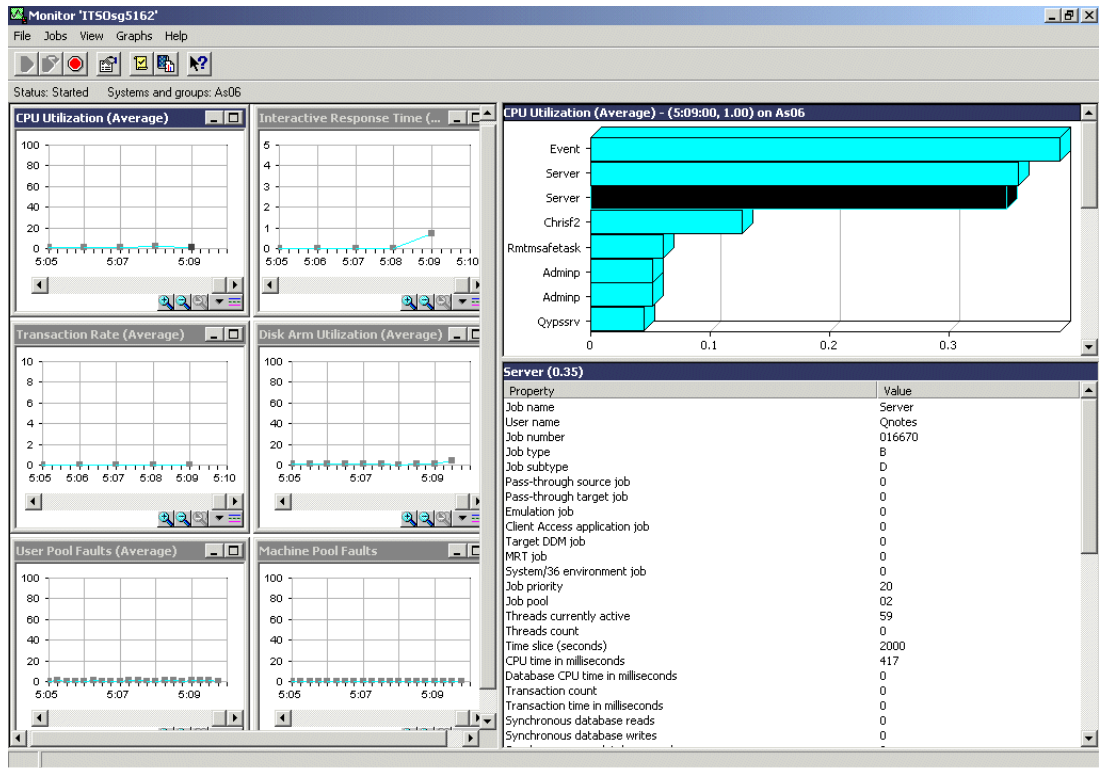


Figure 4-29 Real-time monitor graphical view

The jobs listed in the monitor can be managed and worked on by integrating the Work Management function. For example, you can monitor for the jobs that take up the most CPU, then simply right-click a job, and select **Hold** or **End** and the job will be held or ended immediately.

Figure 4-30 shows an example of how you can hold a job from the real-time monitor view.

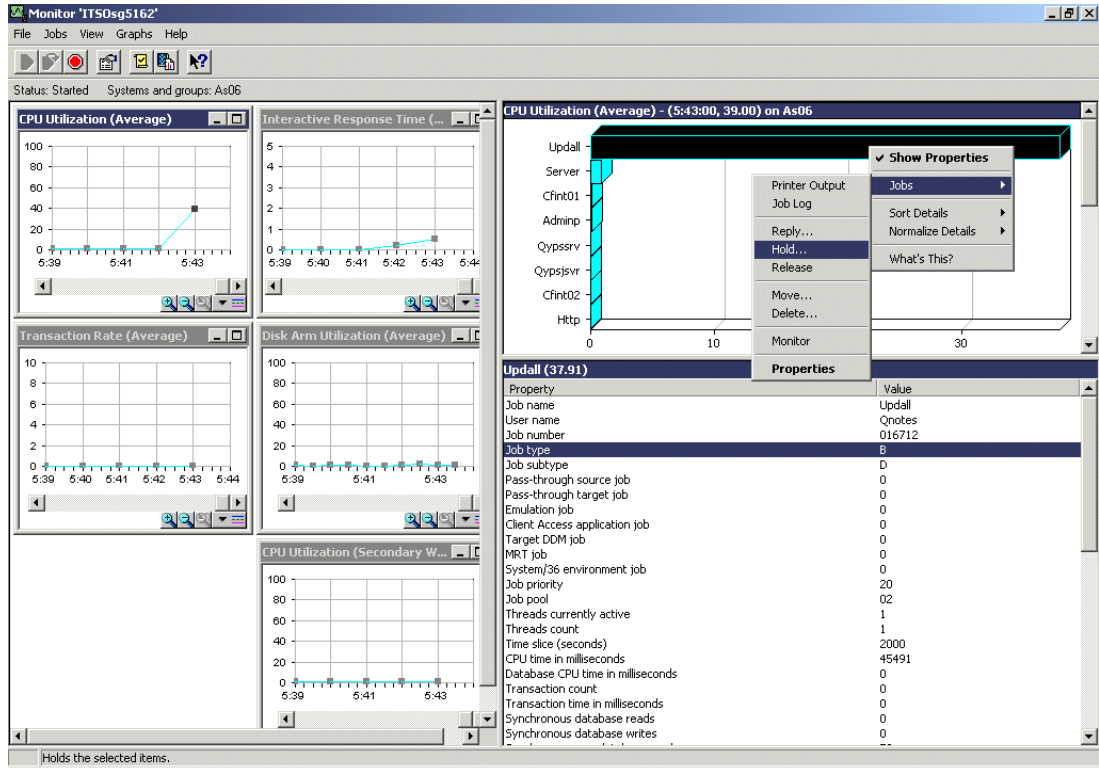


Figure 4-30 Holding a job

Tip: You have more than one server on the system and want to know which server owns the job. In this case, simply right-click a job and select **Job-> Properties-> Tab Active 1**, where you can see which subsystem the job is running under as shown in Figure 4-31.

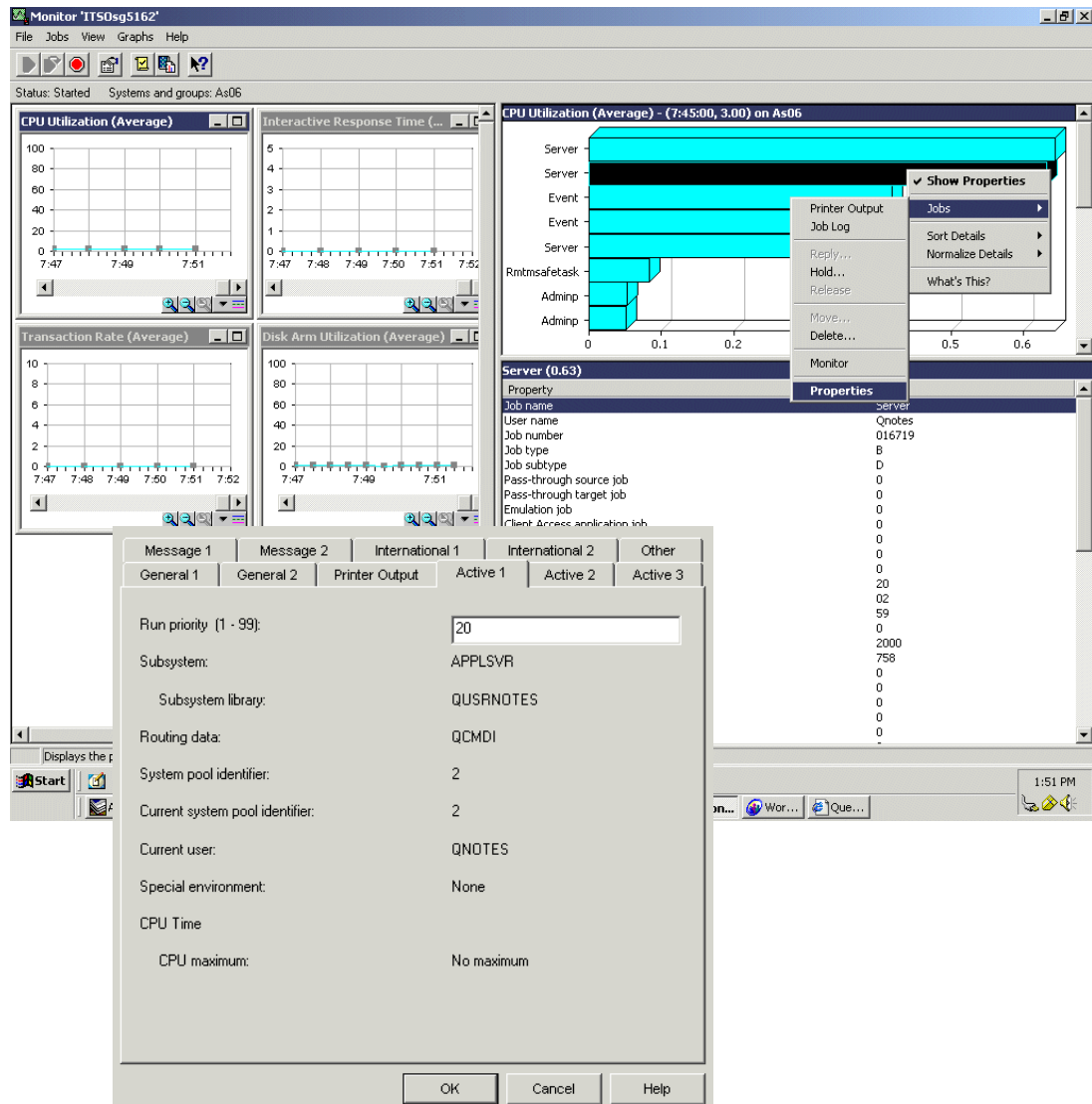


Figure 4-31 Job properties for the server

Monitors also allow you to set and log events based on thresholds. A threshold is defined by the creator of the monitor to be a point at which the operator or an automation routine needs to be involved. For example, upon reaching a threshold, the administrator can define an OS/400 message to be sent to a particular user or a corrective program to be run. When the creator of the monitor uses these thresholds, they can also select to log events. These include two types of events (triggered or reset), and include information such as when, where, and what caused the event.

Figure 4-32 shows an example of the Threshold 1 setting for CPU Utilization (Average). Figure 4-33 shows the monitor graphical view in the background with the Trigger mark for the CPU Utilization (Average) and the Event Log with the Trigger and its properties.

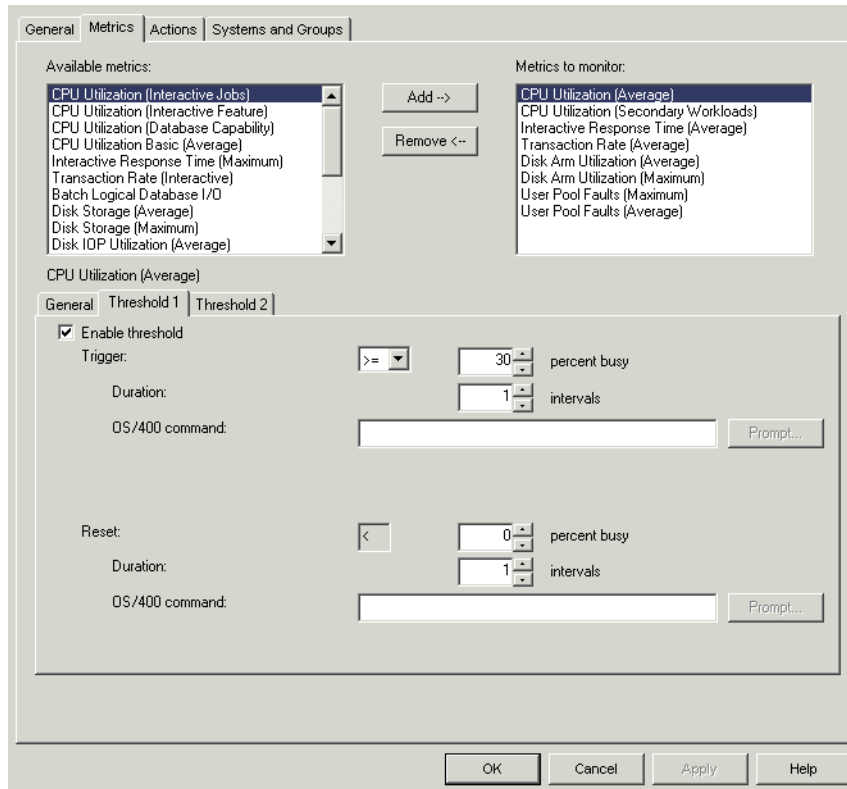


Figure 4-32 Threshold 1 setting for CPU Utilization (Average)

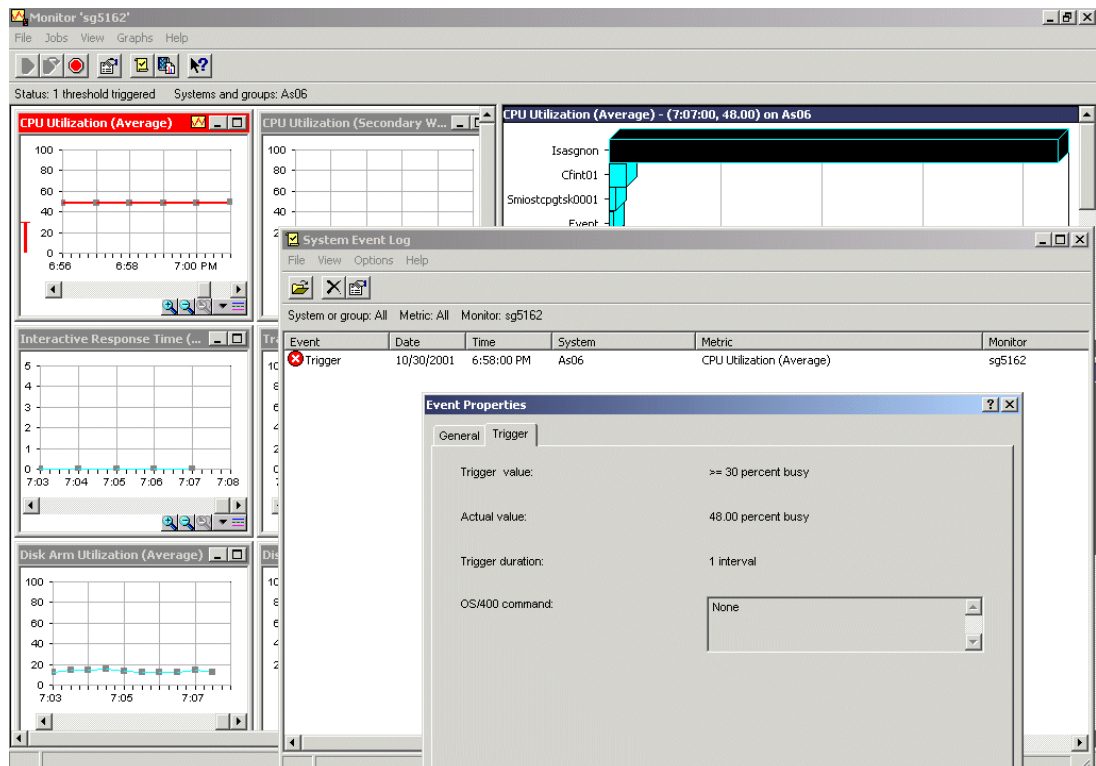


Figure 4-33 Event triggered by Threshold 1 for CPU Utilization (Average)

4.12.2 Viewing graph history of monitor data

In V5R1, you can display a graphical history of performance metrics shown in the system monitors. You can view collected data for the metrics over a long period of time whenever you want, instead of only viewing real-time data. The amount of data that is available to be graphed is determined by the settings that you selected in the Collection Services properties, specifically the Collection retention period.

For more information about Collection Services, see 4.11.1, “Starting Collection Services” on page 83.

If you want to use the Graph History function to see data that was collected days ago, weeks ago, or months ago, you need to have PM/400 activated. PM/400 is presented in 2.5.2, “Performance Management/400” on page 12. Without PM/400 enabled, the graph data field supports one to seven days. With PM/400 enabled, you can define how long your management collection objects remain on the system.

For more information about PM/400, select **File** and **View Trend Analysis** on the Graph History window. This takes you to the PM/400 Web site.

Graph History performance data

The Graph History shows a view of the metrics that you collected for a long period of time for any of the available metrics as compared to the System Monitor window, which shows you real-time data for the last hour. The Graph History window allows you to display one graph at a time. However, you can display more than one Graph History window to make comparisons between systems.

For examples of combining metrics to target specific types of performance information, see 4.12.4, “Combining Management Central’s CPU Utilization metrics” on page 104.

You can view the graph history by following these steps:

1. In Operations Navigator Management Central, you can view the graph history. To do this, right-click either a **system monitor** or a **Collection Services object** and select **Graph History**. Then choose the metric and period you want to view.
2. Click **Refresh** to see the graphical view.
3. To open a new window, select **File->New Graph History Window**.

Figure 4-34 shows an example of the three windows with the Graph History view.

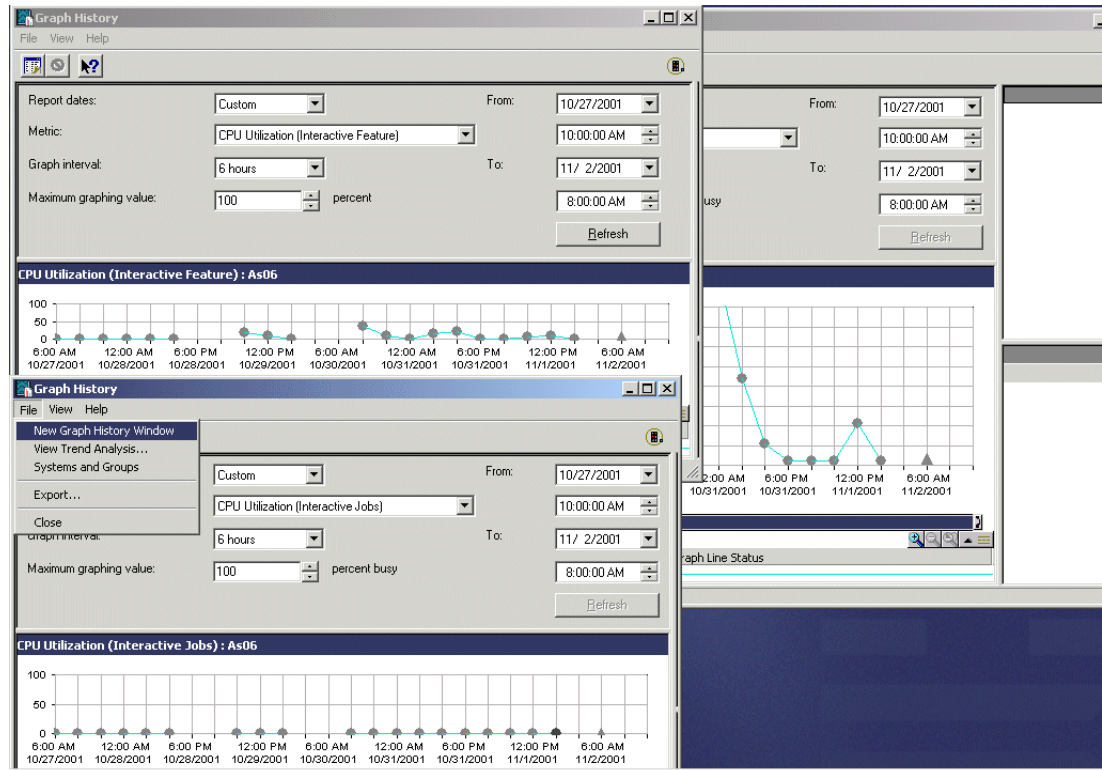


Figure 4-34 Graph History view

Collection points on the graph line are shown by three different graphics that correspond to the three levels of data that are available:

- ▶ A *square* collection point means the data includes both the detailed information and properties information.
- ▶ A *triangular* collection point represents summarized data that contains detailed information.
- ▶ A *circular* collection point represents data that contains no detailed information or properties information.

4.12.3 Exporting Graph History to PC files for analysis and printing

The monitor does not support printing. However, with the new capabilities to view historical data, it can be useful to obtain a printed version of the graph.

The Graph History allows you to export all of the data displayed for the selected metrics, the systems and groups, and the date and time range specified, into a file onto your PC. These PC files provide a history of your data and allow you to work with the data in a spreadsheet program or other application.

The Graph History data can be exported to the following types of PC files:

- ▶ Comma separated variables (.csv)
- ▶ Lotus 1-2-3 compatible (.csv)
- ▶ ASCII tab delimited text (.txt)
- ▶ Microsoft Excel 97 (.xls)
- ▶ Web page (.html)

The export function only exports what is shown in the graph. It would not export all of the data for time periods outside of what is displayed.

Tip: You could manage the retention of output from collection services, using the retention period as permanent. The Graphing/Export capability can then be used to transfer data into a Lotus or Excel spreadsheet and the data can be manipulated to build your own Trend Analyses over multiple collection periods.

4.12.4 Combining Management Central's CPU Utilization metrics

Use Management Central's CPU Utilization metrics (or any other performance metrics) alone or in concert to target specific types of performance information. The information you want to gather from your systems may be more complex than a single metric can provide. Running metrics in combination allows you to analyze system data from multiple perspectives.

Uses for CPU Utilization (Interactive Feature)

One of the best ways to interpret this metric is to use it in combination with others. For example, you might use:

- ▶ *CPU Utilization (Interactive Feature) with CPU Utilization (Average):* This combination gives you data about total system CPU use as well as interactive feature CPU use. This can be helpful in interpreting overall system activity.
- ▶ *CPU Utilization (Interactive Feature) with CPU Utilization (Interactive Jobs):* Jobs of type I are a subset of all jobs included in those analyzed by the CPU Utilization (Interactive Feature) metric. As a result, this combination provides an accurate indication of interactive activity taking place on your system. This also helps you relate your system interactive capability to total CPU capability both at the system and job levels.

Uses for CPU Utilization (Database Capability)

CPU Utilization (Database Capability) can help you identify and resolve the impact of complex queries that are performed while other performance-critical applications are active. You may get the most value out of this specialized metric when you use it in combination with others to understand system database activity relative to other aspects of system activity. For example, you might use:

- ▶ *CPU Utilization (Database Capability) with CPU Utilization (Average):* This combination gives you an accurate perspective of database activity relative to overall CPU activity on your system.
- ▶ *CPU Utilization (Database Capability) in addition to Disk Arm Utilization (Average) or Disk Storage (Average):* This mix of performance metrics provides valuable information in an environment where the affects of database activity are important relative to disk utilization.
- ▶ *CPU Utilization (Database Capability) together with Communication IOP Utilization (Average):* You may choose to use this combination of performance metrics when analyzing a server application (such as Web serving or ODBC) for time spent in communications versus data storage and retrieval.

Uses for CPU Utilization (Secondary Workloads)

Much like the CPU Utilization Basic (Average) metric, CPU Utilization (Secondary Workloads) does not provide job-level detail. You can use CPU Utilization (Secondary Workloads) with other metrics, however, to obtain the additional job-level data if you so require. For example, you can use:

- ▶ *CPU Utilization (Secondary Workloads) with CPU Utilization (Database Capability):* This combination of metrics helps you monitor those Domino applications running ODBC or @DB commands to access DB2 Universal Database for iSeries data.
- ▶ *CPU Utilization (Secondary Workloads) with CPU Utilization (Average), CPU Utilization (Interactive Jobs), and with other similar metrics:* These metrics, when used together, provide a summary of the secondary work taking place on your iSeries server. CPU Utilization (Secondary Workloads) does not provide job-level detail. However, you can use this metric with other metrics to obtain detailed job-level information about activity on your system.

You may discover that different metrics, when used with CPU Utilization (Secondary Workloads), provide related job-level detail that best suits your performance data needs.

4.12.5 Job monitoring

The ability to monitor for jobs across systems was added to Management Central in V5R1, which gives you the possibility to monitor your Domino server jobs running on one or more iSeries servers. For example, you can create a job monitor to monitor one or more systems that provide services for your Domino server applications. Or, you can create a job monitor to monitor batch jobs that run in the background.

The term *job monitor* should not be confused with the term *Performance Monitor* (with OS/400 releases before V5R1), which is now called *Collection Services* (see 4.11, “Collecting performance data” on page 83). Where Collection Services collects performance data typically over a longer period of time, job monitors alert you when the jobs you are interested in do not behave the way you want.

Note: Each V5R1 OS/400 system is shipped with a sample job monitor – Sample Domino Server Monitor. It is provided to assist you in getting started with job monitoring and needs at least a system name added to become operational.

Creating a new job monitor

You can define a new job monitor by using in one of the following methods:

- ▶ Expand **Management Central**. Select **Monitors**. Right-click **Job** and select **New Monitor**.
- ▶ Expand **Basic Operations**. Select **Jobs**. Scan the list of jobs displayed in the right pane and right-click a job. Click **Monitor**. You may need to use the Basic Operations Include function to find the job you are looking for.
- ▶ Expand **Work Management**. Open the appropriate job grouping (Active Jobs, Server Jobs, Job Queues, or Subsystems) to view a list of jobs in the right pane. Right-click a job and select **Monitor**.
- ▶ In Management Central, expand **Monitors**. Select **Job**. Right-click an existing monitor and select **New Based On**, which can save steps in defining a similar monitor from the beginning.

Each of these methods activates the new monitor Properties window shown in Figure 4-35.

General

The General page includes the name of the monitor, a brief description of the monitor, and the jobs and servers to monitor.

On the *Jobs to monitor* subtab, you can specify the jobs to monitor based on any combination of job name, user, subsystem, and job type criteria. You can use an asterisk (*) in the job name, user name, or subsystem name as a substitute for any number of characters, anywhere within the name. Or you can select All to monitor all job names. You can select a job type from the list, or select All to monitor all job types.

Each row in the Selected jobs list shows a combination of criteria that must be met for a job to be monitored. A job is monitored if it meets the criteria in at least one row. Click Add to add the specified criteria to the Selected jobs list. Click Remove to remove a selected row or rows of criteria from the Selected jobs list. Figure 4-35 shows *Server* as the selected job name for user QNOTES in subsystem Domsrv2 to be monitored.

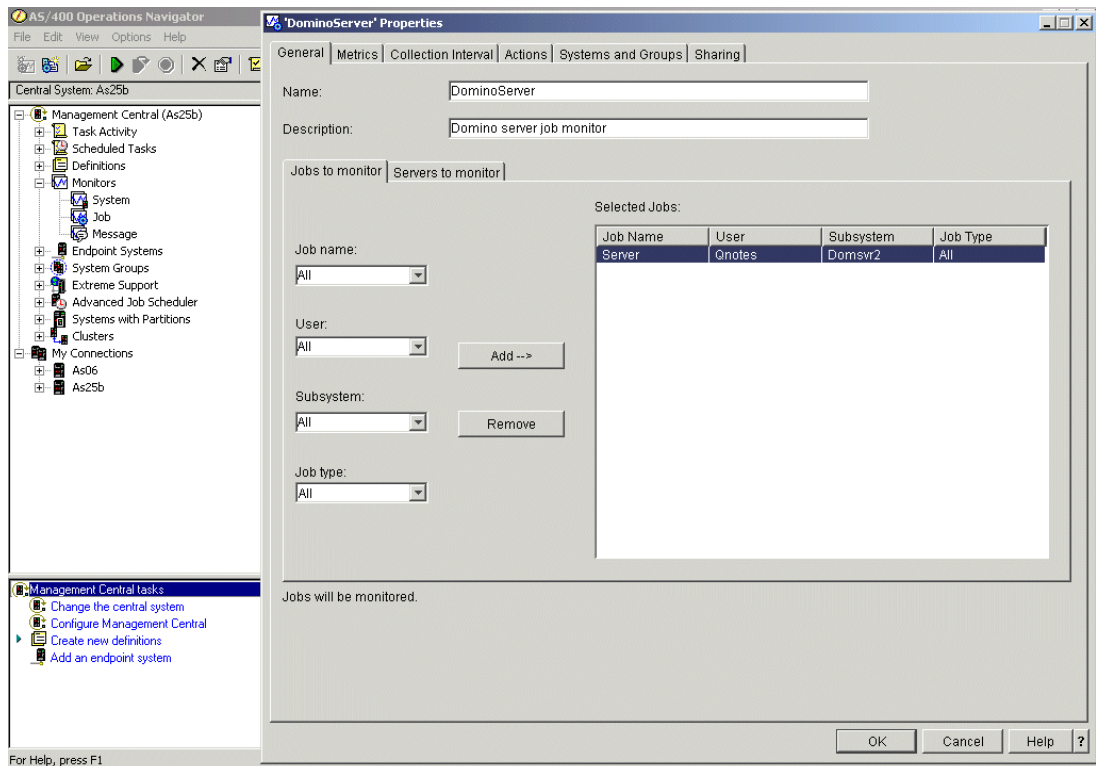


Figure 4-35 Job monitor: Jobs to monitor

On the *Servers to monitor* subtab (see Figure 4-36), you can specify jobs by their server names. Select from the list of available servers on the Servers to monitor page. One of the available servers is the Domino server. If you add that one, you will monitor all Domino server jobs on your system.

Note: The selections you make on the Job to monitor and Servers to monitor pages will be combined. For example, if you select a job with the subsystem name for one of two Domino servers and add Domino servers to be monitored, you will see data from both of the servers.

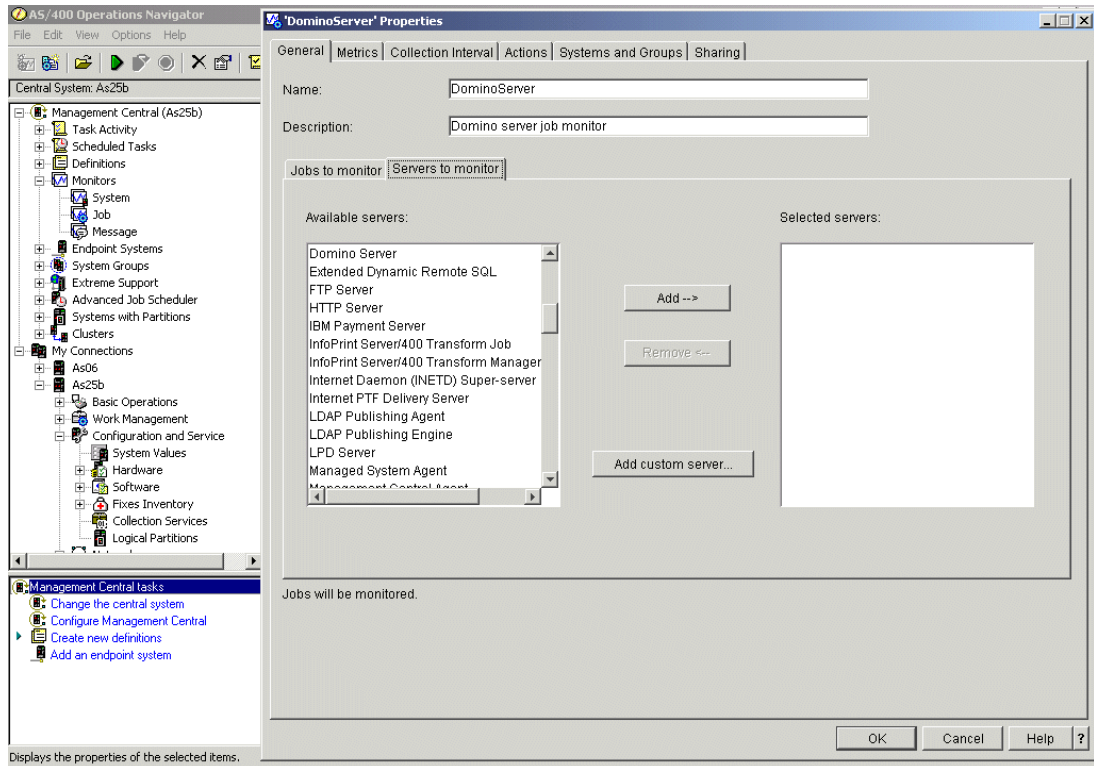


Figure 4-36 Job monitor: Servers to monitor

Metrics

The Metrics page for New Monitor or Monitor Properties allows you to view and change information about each metric to be monitored. You can select metrics to monitor and define thresholds for metric values. The lower portion of the dialog changes based on the selected metric in the Metrics to monitor list. You can define one or two thresholds for each metric. The thresholds are independent of each other and provide a way to monitor for different conditions in one monitor.

For example, you may monitor for a less severe condition and send a command to page the system operator or you may monitor for a more severe condition and send a command page to start ending certain jobs. A threshold consists of a trigger value, and optionally, a reset value. You can specify an OS/400 command to be run when the threshold is triggered or reset.

For each job, the following metrics can be monitored:

- ▶ **Job Count:** Monitor for a specific number of jobs matching the job selection specified under the General tab.
- ▶ **Job Status:** Monitor for jobs in any selected status, such as Completed, Disconnected, Ending, Held while running, or Initial thread held.
- ▶ **Job Log Message:** Monitor for messages based on any combination of Message ID, Type, and Minimum severity.
- ▶ **Job numeric values:**
 - *CPU utilization:* The percentage of available processing unit time used by each job that is being monitored on this system.
 - *Logical I/O rate:* The number of logical I/O actions, per second, by each job that is being monitored on this system.

- *Disk I/O rate*: The average number of I/O operations, per second, performed by each job that is being monitored on this system. The value in this column is the sum of the asynchronous and synchronous disk I/O operations.
- *Communications I/O rate*: The number of communications I/O (read and write) actions, per second, by each job that is being monitored on this system.
- *Transaction rate*: The number of transactions per second by each job that is being monitored on this system. This is meaningful only for 5250 workstation jobs.
- *Transaction time*: The total transaction time for each job that is being monitored on this system. This is meaningful only for 5250 workstation jobs.
- *Thread count*: The number of active threads in each job that is being monitored on this system.
- *Page fault rate*: The average number of times, per second, that an active program in each job, being monitored on this system, refers to an address that is not in main storage.

► **Summary numeric values:**

- *CPU utilization*: The percentage of available processing unit time used by all jobs monitored on this system. For multiple-processor systems, this is the average percent busy for all processors.
- *Logical I/O rate*: The number of logical I/O actions (for example read, write and update) per second, by all jobs monitored on this system.
- *Disk I/O rate*: The average number of physical I/O operations, per second, performed by all jobs monitored on this system. The value in this column is the sum of the asynchronous and synchronous disk I/O operations.
- *Communications I/O rate*: The number of communications I/O (read and write) actions, per second, by all jobs monitored on this system.
- *Transaction rate*: The number of transactions per second by all jobs monitored on this system. This is meaningful only for 5250 workstation jobs.
- *Transaction time*: The total transaction time for all jobs monitored on this system. This is meaningful only for 5250 workstation jobs.
- *Thread count*: The number of active threads for all jobs monitored on this system.
- *Page fault rate*: The average number of times, per second, that active programs in all jobs monitored on this system refer to an address that is not in main storage.

Note: Be careful to monitor the smallest number of metrics that will give you the information you need. Monitoring a large number of metrics may have a negative performance impact on your system.

Figure 4-37 shows an example of the job monitor Metrics tab, where CPU Percent Utilization is monitored and will create a trigger if CPU% is > (greater than) 10%.

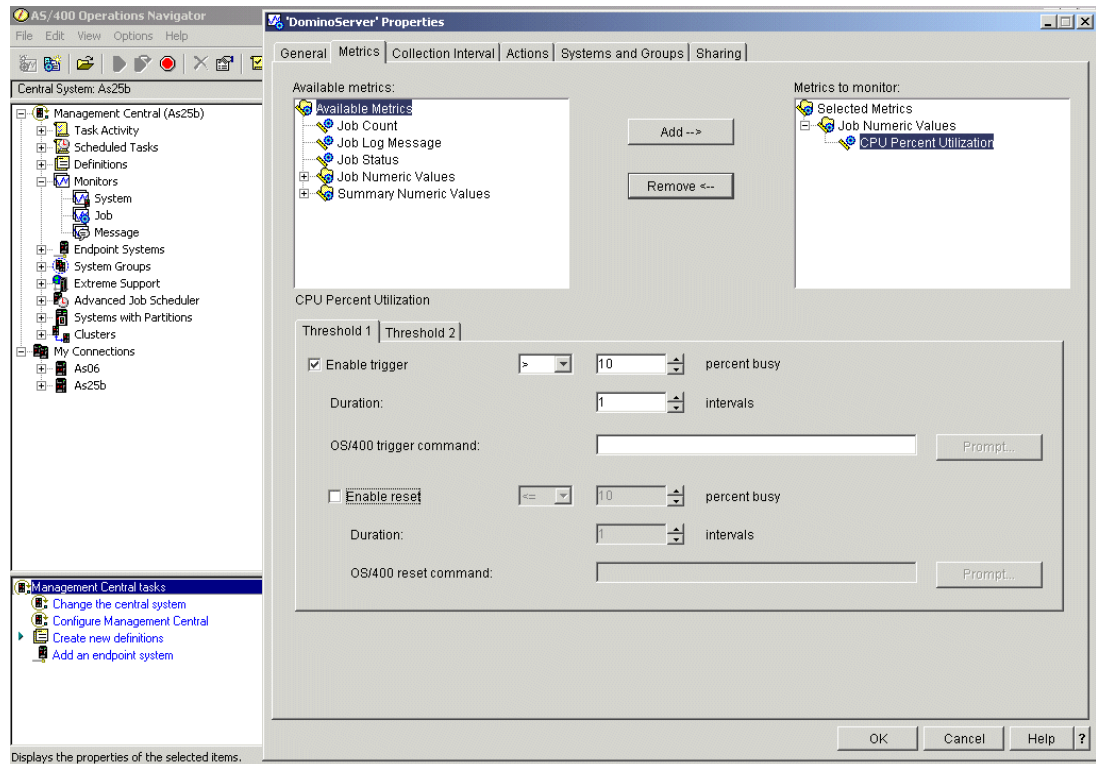


Figure 4-37 Job monitor: Metrics

Collection Interval

When you are setting thresholds for the metrics you have selected to monitor, you should consider how often you want the data to be collected or sampled. For example, sampling every 5 minutes or even 15 minutes might be right for your environment.

You can use the same or different collection interval values for each metric. This interval works in conjunction with the Duration parameter under Threshold on the Metrics page. That is, you specify the time interval value on the Collection Interval page, such as 5 minutes. For Duration on the Metrics page, you specify two intervals. This means the threshold condition is examined every 5 minutes, but the condition must last for two intervals or 10 minutes.

If you want to monitor numeric and status metrics for less than 5 minutes, you must select *Use different collection interval for each metric type*.

Figure 4-38 shows an example of the job monitor Collection Interval tab set to 5 minutes.

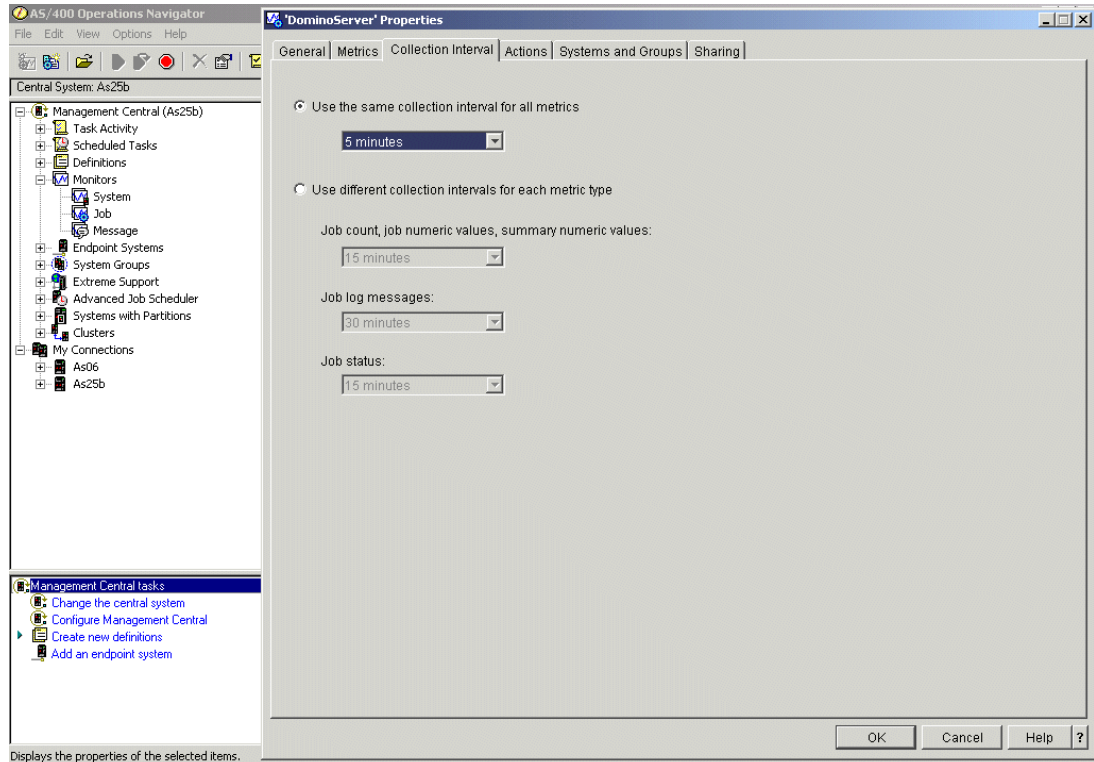


Figure 4-38 Job monitor: Collection Interval

Actions

When you have specified the threshold values for your job monitor, you can click the Actions tab to select event logging and PC actions to be taken when a threshold is triggered or reset.

The actions that you can select are:

- ▶ **Log event:** Adds an entry to the event log on the central system when the threshold is triggered or reset. The entry includes the date and time the event occurred, the endpoint system being monitored, the metric being collected, and the monitor that logged the event.
- ▶ **Open event log:** Displays the event log when a trigger or reset event occurs. This automatically brings up the Event Log window if it is not currently displayed on PC.
- ▶ **Open monitor:** Displays a list of systems that are being monitored for the specified metrics and a list of the values for the specified metrics as they are collected for each system when a trigger or reset event occurs. This automatically brings up the Monitor window if it is not currently displayed on the PC.
- ▶ **Sound alarm:** Sounds an alarm on the PC when the threshold for the monitor is triggered.
- ▶ **Run OS/400 command:** A check mark indicates whether a command has been defined for a threshold triggered or reset. If you have specified an OS/400 command to run when the threshold for this monitor is triggered or reset, those commands run only during times that actions are applied. This option cannot be changed from the Actions page. If you do not want the command to run, you can remove the command from the Metrics page.

In the example shown in Figure 4-39, the Apply thresholds and actions and when to apply values are modified.

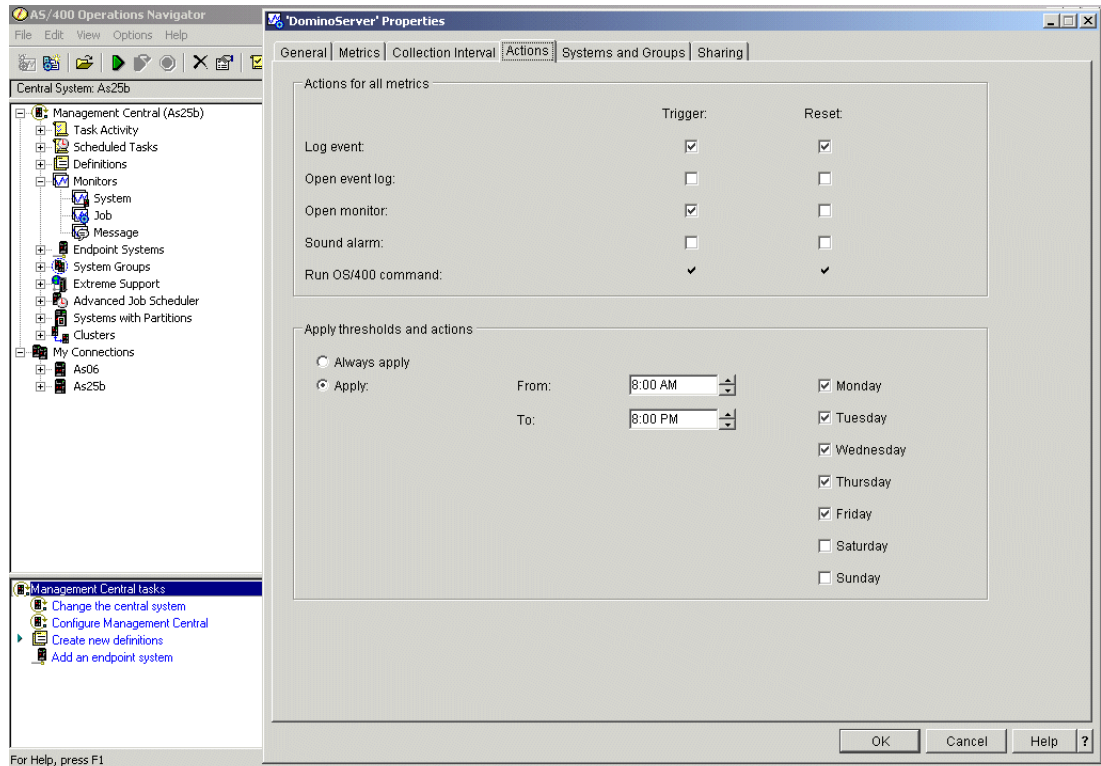


Figure 4-39 Job monitor: Action

Systems and Groups

Use the Systems and Groups page to view and change the endpoint systems and groups on which you want the job monitor to run. When you start the monitor, the default system or systems to start on will be those specified here. If the activity has already been started, changes made on this page become active when you click OK. The activity is stopped on endpoint systems that are no longer in the list. The activity is started on systems that you added to the list.

Figure 4-40 shows an example of the job monitor Systems and Groups display.

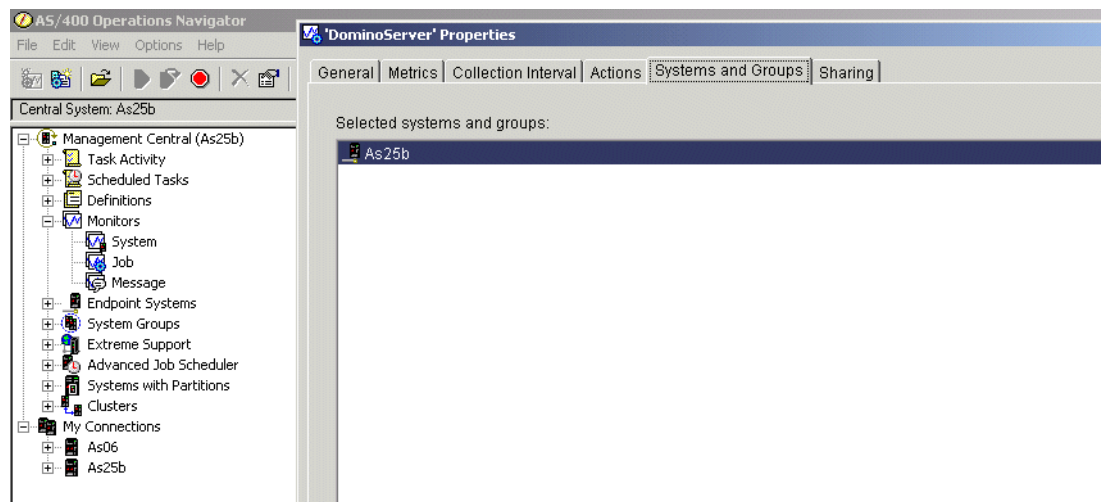


Figure 4-40 Job monitor: Systems and Groups

Sharing

This is very important, depending on whether you want other OS/400 users to view and manage your monitor. The sharing options that are available to the creator of the monitor are:

- ▶ **None:** Other users cannot view this monitor.
- ▶ **Read-Only:** Other users can view and use this monitor, but cannot make any changes to it. They can, however, create a new monitor based on this one and specify their own share values on that monitor.
- ▶ **Controlled:** Other users can stop, start, and view this monitor and use it as a template for a new monitor. Only the owner can change the level of sharing, and others cannot change any of the monitors parameters.

Starting a job monitor

When your monitor is defined, it appears in Management Central under its specific job monitor type. Right-click the named monitor and select **Start**.

Just as for a system monitor, once you start a job monitor, you are free to do other tasks on your server, in Operations Navigator, or on your PC. In fact, you could even turn off your PC, in which case it would continue to monitor your systems and perform any threshold commands or actions you specified. Your monitor will run until you decide to stop it. Figure 4-41 shows an example of how you can start job monitor.

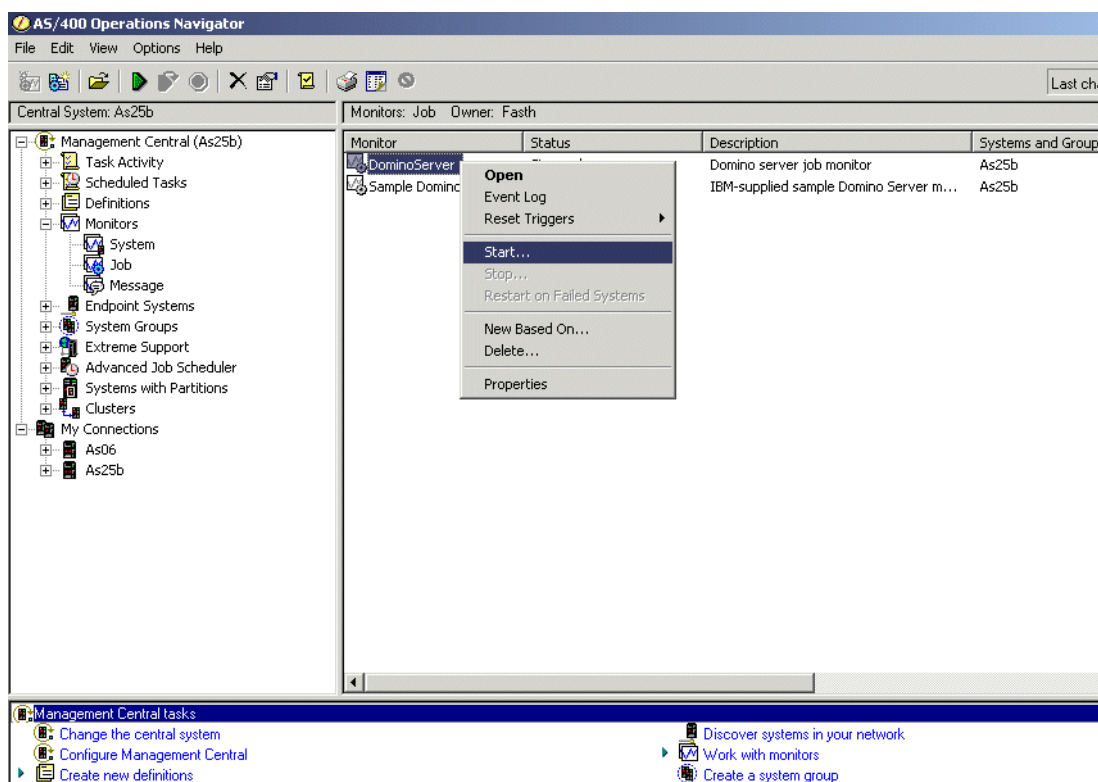


Figure 4-41 Starting a job monitor

Viewing job monitor results

After you have defined and started your monitor, you are ready to view your monitor results. Double-click the monitor name to open the Job Monitor window. In the Job Monitor window, you can see the overall status of the monitor and the target systems on which the monitor is running.

In the example shown in Figure 4-42, we defined the monitor for the server job in subsystem Domsrv2 and specified a threshold for the server job that averages 10 percent of CPU over a five minute period.

Tip: In the Job monitor view, you can select Options and Columns F11 to customize your view. For example, if you have more than one Domino server on the system, include the subsystem name and you can see which Domino server owns the server jobs.

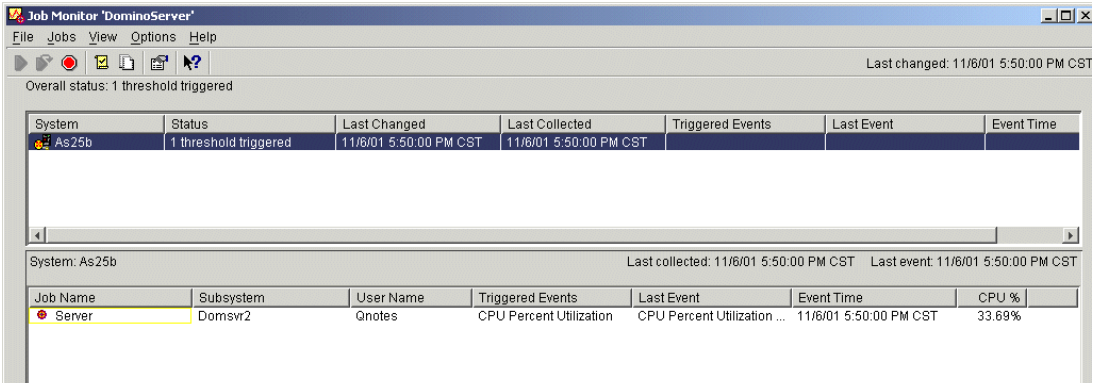


Figure 4-42 Job monitoring CPU percent utilization

Viewing a job monitor event log

The Event Log window displays a list of threshold trigger and reset events for job monitors. In our example in Figure 4-39, we specified under the Action tab that we want to add events to the Event log. As the server job reached CPU 33.69%, an event was added to the Event Log, as shown in Figure 4-43.

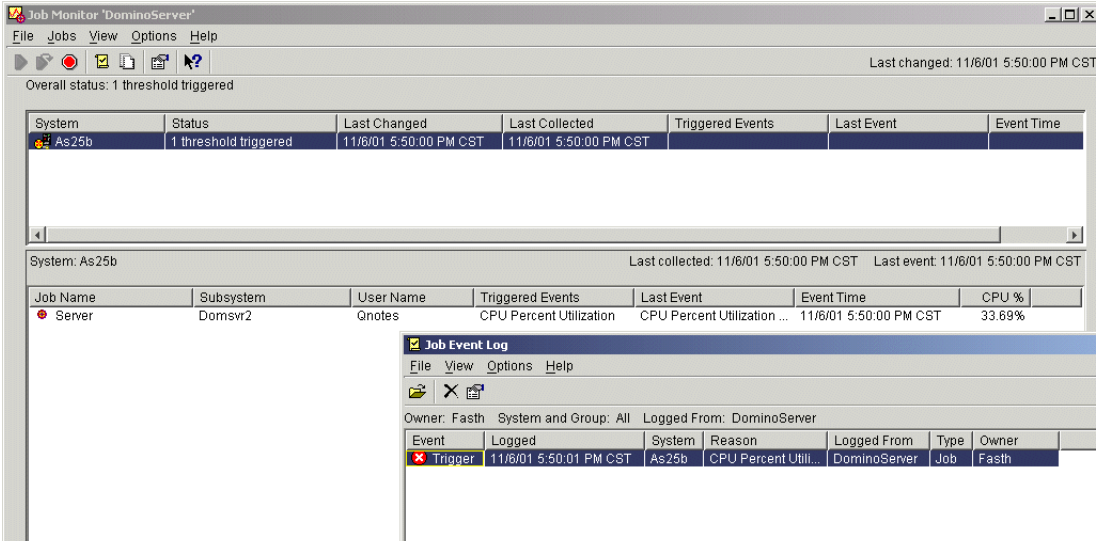


Figure 4-43 Job Monitor and Event Log

4.13 Reviewing data using Performance Tools/400

Performance Tools for iSeries Licensed Program Product (5722-PT1) provides the ability to:

- ▶ Summarize the performance data into reports
- ▶ Analyze performance data
- ▶ Perform system modeling based on performance data

Refer to the following publications for a description of these and other capabilities of the performance tools and how to use them:

- ▶ *Performance Tools for iSeries*, SC41-5340
- ▶ *BEST/1 Capacity Planning Tool*, SC41-5341

4.13.1 Print System Report (PRTSYSRPT) command

The PRTSYSRPT command creates and prints a system operation overview report from the performance data collected by Collection Services. The report is written to the printer file QPPTSYSR. The system workload, resource utilization, storage pool utilization, disk utilization, and communications summary are shown in the report.

4.13.2 Print Component Report (PRTCPTTRPT) command

The PRTCPTTRPT command creates a report that provides more detailed information than in the System Report. Detailed information regarding the system performance of each component is reported.

This detailed report is produced from the performance data collected by Collection Services and shows the data by job, user, pool, disk, IOP, local workstation, and exception. The report is written to the printer file QPPTCPTTR. Jobs can be selectively included in the report or excluded from the report, based on a variety of job details and interval times.

4.13.3 Print Transaction Report (PRTTNSRPT) command

The PRTTNSRPT command creates and prints performance reports that show detailed information about the transactions that occurred during the time that the performance data was collected. These reports require that trace data is collected. To collect trace data, use the STRPFTRRC command (Figure 4-44). You should selectively include or exclude jobs from the reports, based on a variety of job details and interval times.

Start Performance Trace (STRPFRTRC)

Type choices, press Enter.

Size	*CALC	128-998000, *CALC, *MAX
Omit trace points	*NONE	*NONE, *RSCMGT

Additional Parameters

Job types	*DFT	*NONE, *ALL, *DFT, *ASJ...
+ for more values		
Job trace interval	0.5	.1 - 9.9 seconds

Bottom

F3=Exit F4=Prompt F5=Refresh F12=Cancel F13=How to use this display
F24=More keys

Figure 4-44 STRPFRTRC command parameters

Note: The Transaction Report provides Domino-related information down to the service program level only. At the time this redbook was written, none of the tools provided information at the module level.

Change Job Type (CHGJOBTYP) command

Use the CHGJOBTYP command to alter the job type for jobs that appear on the reports produced by using the PRTTNSRPT command. For example, use this command to change the SERVER job type from batch to interactive to get statistics about the number of transactions on the Domino server.

Note: The number of transactions obtained with this approach is not the same value that you find with using Domino Statistics and Logs. However, it may assist you by providing a number that represents an amount of work. Then, you can use that number for growth planning purposes.

4.13.4 Print Activity Report (PRTACTRPT) command

The PRTACTRPT command creates reports based on the data collected by the Work With System Activity (WRKSYSACT) command.

4.13.5 Convert Performance Thread Data (CVTPFRTHD) command

The CVTPFRTHD command converts performance data records collected by Collection Services. The specified member (MBR parameter) of the QAPMJOBL database file contains records with thread-level performance data. Use the CVTPFRTHD command to convert this data and write the resulting records to a member by the same name (MBR parameter) in the QAPMTJOB file. The output file member contains records with job-level performance data, which is a total of the performance information for all threads running within the job.

The input file (QAPMJOBL) must exist in the library specified on the LIB parameter. If the QAPMTJOB file does not exist in the specified library, it is created automatically. A file member by the name specified (MBR parameter) is automatically added to file QAPMTJOB if it does not already exist.

4.14 Performance Explorer (PEX)

Performance Explorer is a tool that helps find the causes of performance problems that cannot be identified by using tools that do general performance monitoring. Using the Performance Explorer requires more specialized knowledge. You need to:

- ▶ Have an idea of where the problem may be on the iSeries server
- ▶ Set up PEX to collect data in specific areas
- ▶ Interpret the PEX data

For more information on the Performance Explorer, refer to the following resources:

- ▶ *Performance Tools for iSeries*, SC41-5340
- ▶ *AS/400 Performance Explorer Tips and Techniques*, SG24-4781
- ▶ iDoctor for iSeries site at: http://www.as400service.ibm.com/i_dir/idoctor.nsf

You can find another example on using PEX in 12.5, “Observing Domino processing with Performance Explorer” on page 365.

Performance Explorer is probably the last tool that you use to analyze a performance problem. Using PEX requires some special skills. It is not the tool with which to start your analysis. The following list shows the process (or the cycle) for using the PEX tool:

1. Add the PEX Definition (ADDPEXDFN).
2. Start Performance Explorer (STRPEX).
3. End Performance Explorer (ENDPEX).
4. Print the PEX Report (PRTPEXRPT).

4.14.1 Add PEX Definition (ADDPEXDFN) command

The ADDPEXDFN command allows you to add a new Performance Explorer definition to the system. Each definition is stored as a member in the QAPEXDFN file in library QUSRSYS. A Performance Explorer definition identifies the performance data that is collected during a Performance Explorer session. A session can be started using the Start Performance Explorer (STRPEX) command. When starting a new session, a Performance Explorer definition name is required.

4.14.2 Start PEX Session (STRPEX) command

The STRPEX command allows you to start a new Performance Explorer session. Or, you can resume a previously suspended Performance Explorer session.

4.14.3 End PEX Session (ENDPEX) command

The ENDPEX command allows you to stop collecting data. The command requires a session name to accompany the request that identifies which instance of the Performance Explorer session to end.

You can either end the data collection session or suspend the data collection session. If you choose to end the session, the collected data is put into a single physical file or multiple database files, or it is deleted, based on the value specified for the data option (DTAOPT) parameter.

If you choose to suspend the collection of performance data, the session remains active. To resume data collection for a suspended session, specify OPTION(*RESUME) on a subsequent call of the STRPEX command.

4.14.4 Print PEX Report (PRTPEXRPT) command

The PRTPEXRPT command allows you to print a formatted listing of the data that is collected by the Performance Explorer and is saved across a set of physical files in a particular library.

Restriction: You must have read authority for each Performance Explorer database file in the specified library.

4.14.5 Performance Explorer trace points for Domino

Starting with Domino 5.06a and OS/400 V4R4, Domino has added trace points to further assist in resolving performance-related problems that cannot be seen using other tools. There are currently six trace points. The trace points are enabled through the notes.ini file and the data is collected using PEX.

PEX trace points for Domino

The PEX trace points are enabled by adding a line, DEBUG_OS400_PEX with a value equal to 1, 2, 3, or 4 to the notes.ini file. It enables the PEX trace points with varying degrees of collection information, 1 being the least amount of data collected and 4 being the most data.

The trace point structure is designed to collect data for the selected level along with levels less than the selected value. For example, setting DEBUG_OS400_PEX=2 collects data for the QIBM_QLNT_OPENDB, QIBM_QLNT_OPENCOLL, QIBM_QLNT_UPDATECOLL, and QIBM_QLNT_HTTP_URL trace points. This structure was implemented to reduce the amount of data that can be logged as a result of using QIBM_QLNT_OPENDBALL.

Table 4-3 describes the six trace points and the information they provide when enabled.

Table 4-3 PEX trace points for Domino

Trace point name	Level	Type of information collected
QIBM_QLNT_OPENDB	1	Databases opened from the Notes Client. Logs the database name, user name, and the user's IP address.
QIBM_QLNT_OPENCOLL	1	Collection (view) opened from the Notes Client. Logs the database name, view name, user name, and the user's IP address.
QIBM_QLNT_UPDATECOLL	1	Collection (view) refreshed or updated from the Notes Client. Logs the database name, view name, user name, and the user's IP address.

Trace point name	Level	Type of information collected
QIBM_QLNT_HTTP_URL	2	URLs that are received by the HTTP server. This includes HTML pages, images, etc. You will see multiple entries for one page request due to sending all the page elements such as images. It also logs the URL requested.
QIBM_QLNT_SVR_AGENT	3	Scheduled Server Agents that are ran through AMGR. Logs the database name, agent name, and the time the agent took to complete.
QIBM_QLNT_OPENDBALL	4	Database opens internally in Domino. This will log many more opens than just QIBM_QLNT_OPENDB since it logs opens that Domino does internally. This can be used to see what databases Web clients or agents may be accessing. However, every open will not be logged here since every open does not pass through this trace point. If this is enabled, then your trace will likely consist of mostly these trace points. Logs the database name.

Installing the PEX trace points for Domino

For releases prior to V5R1, PTFs are required to install the trace points for Domino. Table 4-4 indicates the PTFs for each release of OS/400 that are needed.

Table 4-4 PTFs required to install PEX trace points for Domino

OS400 version and release	PTF
V4R4	SF64028*
V4R5	SF64108
V5R1 and later	No PTF is required.
* These PTFs can be applied immediately and do not have any activation or special instructions.	

Enabling the PEX trace points

Enabling PEX trace points is accomplished by adding a line to the notes.ini file. To do this, you add DEBUG_OS400_PEX=X, where X equals the desired collection level. The new line can be added anywhere in the file.

To modify the notes.ini file, use the Work with Domino Servers (WRKDOMSVR) command and choose option 13 (Edit NOTES.INI) next to the applicable Domino server (Figure 4-45).


```

Work with Domino Servers
System:  ACME01

Type options, press Enter.
1=Start server    2=Change server    5=Display console    6=End server
7=Submit command  8=Work console    9=Work server jobs
11=Change current directory  12=Work object links  13=Edit NOTES.INI

    Domino
Opt  Server      Subsystem      Domino
    *HTTPSETUP    QDOMINOHT      Status
    MAILDOMXX     MAILDOMXX      *ENDED
    MAILDOM02     MAILDOM02      *ENDED
13  MAILSVR      MAILSVR        *STARTED
    APPLSVR      APPLSVR        *STARTED

Bottom

Parameters or command
===>
F3=Exit    F4=Prompt    F5=Refresh    F9=Retrieve    F11=Display path
F12=Cancel  F17=Top    F18=Bottom    F22=Display entire field    F24=More keys

```

Figure 4-45 Work with Domino Servers command

Figure 4-46 shows the editing view for the notes.ini file that appears next. Add `DEBUG_OS400_PEX` to the bottom of the notes.ini file with the desired logging level. Once complete press F3 twice to save and exit this screen. The newly added changes will not take effect until the Domino server has been stopped and started or you've performed the Domino command `RESTART SERVER`.

```

Edit File: /domino/mailsvr/NOTES.INI
Record :   80 of   91 by 10      Column :   1   59 by 126
Control :

CMD
....+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8....+...
.9....+....0....+....1....+....2....+..
TRANSLLOG_AutoFixup=1
TRANSLLOG_UseAll=0
TRANSLLOG_Style=0
TRANSLLOG_Performance=2
TRANSLLOG_Status=1
MTEnabled=0
SCHEDULE_VERSION=3
WebAdminSetup=503
DominoConfigLevel=3
TRANSLLOG_MaxSize=1028
TRANSLLOG_Path=logdir
DEBUG_OS400_PEX=2
*****End of Data*****

F2=Save  F3=Save/Exit  F12=Exit  F15=Services  F16=Repeat find  F17=Repeat change
F19=Left  F20=Right

```

Figure 4-46 Adding `DEBUG_OS400_PEX` to notes.ini

After all necessary collections are made, go back into the notes.ini file and comment out DEBUG_OS400_PEX until it is needed again. To comment out this entry, place a semicolon in front of the line as shown in Figure 4-47.

```

Edit File: /domino/mailsvr/NOTES.INI
Record : 80 of 91 by 10          Column : 1 59 by 126
Control :

CMD
....+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8....+...
.9....+....0....+....1....+....2....+..

DominoConfigLevel=3
TRANSLLOG_MaxSize=1028
TRANSLLOG_Path=logdir
;DEBUG_OS400_PEX=2
*****End of Data*****

F2=Save F3=Save/Exit F12=Exit F15=Services F16=Repeat find F17=Repeat change
F19=Left F20=Right

```

Figure 4-47 Commenting out DEBUG_OS400_PEX

Note: Adding and commenting out DEBUG_OS400_PEX requires that you stop and start the Domino server before the changes will take effect.

Creating a PEX definition for Domino

A PEX definition identifies what data needs to be collected. Depending on the Domino and the OS/400 release, one of three commands shown in Table 4-5 need to be performed to add the PEX definition as a member to file QAPEXDFN in library QUSRSYS.

Table 4-5 ADDPEXDFN commands for V5R1, V4R5, and V4R4

OS/400 VRM	Command
V5R1	ADDPEXDFN DFN(DOMINO) TYPE(*TRACE) JOB(*ALL) MAXSTG(100000) TRCTYPE(*SLTEVT) SLTEVT(*YES) OSEVT((*DOMTRCDA))
V4R5	ADDPEXDFN DFN(DOMINO) TYPE(*TRACE) JOB(*ALL) MAXSTG(100000) TRCTYPE(*SLTEVT) SLTEVT(*YES) OSEVT(*MIEV12)
V4R4	ADDPEXDFN DFN(DOMINO) TYPE(*TRACE) JOB(*ALL) MAXSTG(100000) TRCTYPE(*SLTEVT) SLTEVT(*YES) OSEVT(*MIEV12)

Executing one of these commands creates a PEX definition called *Domino*. The definition name does *not* have to be *Domino*. It can be called any name.

Collecting a PEX trace for Domino in its entirety

We discussed the necessary components needed to run a PEX trace on Domino. This section walks you through making the collection:

1. Add DEBUG_OS400_PEX with the desired level of collection to the notes.ini file. See “Enabling the PEX trace points” on page 118 for further information.

Important: You *must* end and restart the Domino server for this to take effect.

2. Create a library for the data to be dumped into when you end the trace. To do so, use the Create Library command:

```
CRTLIB LIB(MYLIB)
```

The library's name can be a name of your choice.

3. Add the desired PEX definition based on the level of OS/400 you are running on. See "Creating a PEX definition for Domino" on page 120 for further information.
4. Start the PEX trace using the following command:

```
STRPEX SSNID(MYTRACE) DFN(DOMINO)
```

Note: For the SSNID (Session ID) parameter of the STRPEX command, you can call the session name any name you choose.

The PEX trace is now active. To verify whether any events have been logged, use the ENDPEX command and press Enter. Figure 4-48 shows an example of an ENDPEX display. On the far right-hand side of the screen, you see the Event Count column. This column indicates the number of events that have been collected in the active trace so far.

Attention: If the iDoctor for iSeries PEX Analyzer tool is installed on your system, you could use the QYPINT/STRIDOCOL command to automatically start and end the PEX collection for you. Simply put the name of the definition in the PROBLEM parameter, the length you want the trace to run in the DURATION parameter, and change the PRTPEX parameter to "yes". This command does all the work for you. It starts and ends the PEX trace along with printing the PEX data.

If you do not have the iDoctor for iSeries PEX Analyzer installed, but want more information on the tool, go to: http://www.as400service.ibm.com/i_dir/idoctor.nsf

Select Performance Explorer Session					
Type option, press Enter.					
1=Select					
Option	Session	User	Type	State	Event count
	MYTRACE	MYUSER	*TRACE	ACTIVE	29311
					Bottom
F3=Exit	F5=Refresh	F12=Cancel			

Figure 4-48 ENDPEX display

On average, 100 mail users collect 10M worth of raw PEX data in 15 minutes, so keep this in mind when deciding whether to end an active PEX collection. Also note that based on the current PEX definition we defined, we can collect up to 100 MB worth of PEX trace data before the trace goes into a suspended state. If a larger data collection size is needed, you can increase the MAXDATA parameter on the ADDPEXDFN command up to 4 GB at V5R1.

- Ending PEX can be a long running process and since a good portion of the machines in production today are iSeries servers with a limited amount of interactive workload capability, we recommend that you end PEX by submitting a batch job as shown here:

```
SBMJOB CMD(ENDPEX SSNID(MYTRACE) DTALIB(MYLIB)) JOB(ENDPEX)
```

Based on this example, you now have a batch job running called ENDPEX. This job must complete before you continue on to step 6 and should not be ended manually at anytime, or the PEX data will be lost.

- Since the ENDPEX job has finished, you are now ready to print the data using the PRTPEXRPT command. This also can be a long running job. Therefore, it should be submitted as a batch job to print the PEX data as shown here:

```
SBMJOB CMD(PRTPEXRPT MBR(MYTRACE) LIB(MYLIB) TYPE(*TRACE)) JOB(PRTPEXRPT)
```

Once the PRTPEXRPT job finishes, it produces a spooled file named QPVPERPT. The information in the spooled file resembles the data shown in Figure 4-49. The spooled file is 150 columns wide so not all the information is viewable on the screen at one time.

Display Spooled File															Page/Line 8/17					
File : QPVPERPT															Columns 1 - 130					
Control																				
Find																				
*...+...1...+...2...+...3...+...4...+...5...+...6...+...7...+...8...+...9...+...0...+...1...+...2...+...3																				
		Address	Offset	Object Name	Obj	Seg	PRE	NPgs	LIC-Pgm--	Offset	MI-Pgm----	Offset	NAGP	PI						
					T	ST	T	ST	FIX						EXID	IE				
ss.mmm	P	M	Task ID	Parent-Pgm	HLL-No	CurrentPgm	RC	Delta	Run	Cycles	Event	PTY	Unit	Sector	Span	SKP XCH	WaitSleep	Cycles	WODSC	RS
34.635	00	0	0000000015F1					0	587865		MEV12	API	QIBM_QLNT_HTTP_URL	0			00**/homepage.ns			
34.637	00	0	0000000015F1					223433	811298		MEV12	API	QIBM_QLNT_OPENDB_ALLO				00**/xenon/notes			
34.658	00	0	0000000015F1					2696295	3507593		MEV12	API	QIBM_QLNT_OPENDB_ALLO				00**names.nsf			
34.674	00	0	0000000015F1					1953123	5460716		MEV12	API	QIBM_QLNT_OPENDB_ALLO				00**/xenon/notes			
34.686	00	0	0000000015F1					1504802	6965518		MEV12	API	QIBM_QLNT_OPENDB_ALLO				00**names.nsf			
34.687	00	0	0000000015F1					178390	7143908		MEV12	API	QIBM_QLNT_OPENDB_ALLO				00**/xenon/notes			
34.700	00	0	0000000015F1					1580501	8724409		MEV12	API	QIBM_QLNT_OPENDB_ALLO				00**/xenon/notes			
34.790	00	0	0000000015F2					0	369295		MEV12	API	QIBM_QLNT_HTTP_URL	0			00**/homepage.ns			
34.792	00	0	0000000015F2					141695	510990		MEV12	API	QIBM_QLNT_OPENDB_ALLO				00**/xenon/notes			
34.794	00	0	0000000015F3					0	385495		MEV12	API	QIBM_QLNT_HTTP_URL	0			00**/xenon/notes			
34.827	00	0	0000000015F4					0	340035		MEV12	API	QIBM_QLNT_HTTP_URL	0			00**/homepage.ns			
34.828	00	0	0000000015F4					134818	474853		MEV12	API	QIBM_QLNT_OPENDB_ALLO				00**/xenon/notes			
35.096	00	0	0000000015F2					0	1458298		MEV12	API	QIBM_QLNT_OPENDB_ALLO				00**/xenon/notes			
35.115	00	0	0000000015F4					0	1392245		MEV12	API	QIBM_QLNT_OPENDB_ALLO				00**/xenon/notes			
43.110	00	0	0000000015DD					0	233407		MEV12	API	QIBM_QLNT_OPENDB_ALLO				00**/xenon/notes			
															More...					
F3=Exit F12=Cancel F19=Left F20=Right F24=More keys																				

Figure 4-49 PRTPEXRPT spooled file

- Obviously, analyzing the data is the most important part of the entire process. All the necessary information to analyze the problem is contained within the spooled output (QPVPERPT) created from running the PRTPEXRPT command. If further analysis is needed, the PEX data is stored in DB2 files so SQL and Query/400 can be used to extrapolate the data further. Example 4-1 shows an SQL statement that can be used over the PEX DB2 files to further extrapolate the data. The iDoctor for iSeries, a new function provided for OS/400 V5R1, allows you to further examine the results of PEX Analyzer. For more information on iDoctor, see http://www.as400service.ibm.com/i_dir/idoctor.nsf

Example 4-1 SQL statement that can be used over the DB2 files

```
select qayperuni.qrnts+qaypetidx.qtitim microseconds,
       qaypemiusr.qrecn, qmudta, qaypetaski.qtsjnm, qtsjus, qtsjnb
from qaypemiusr, qaypetidx, qayperuni, qaypetaski
```

where (qaypemiusr.qrecn = qaypetidx.qrecn) and (qaypetaski.qtsctn =
qaypetidx.qtitct)
order by qaypemiusr.qrecn

Figure 4-50 shows a view of how the SQL statement in Example 4-1 would appear if it was run against PEX data. Further information containing the job name, job user, and job number is shown to the right of the information shown in Figure 4-50.

Display Data					Data width : 316												
Position to line					Shift to column												
....+....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8....+....9....+....10....+....11....+....12....+....13.																	
Timestamp expression	Rec No	User	data														
2001-03-01-12.57.34.634696	1	API	QIBM_QLNT_HTTP_URL	0	* 00**/homepage.nsf?Open												
2001-03-01-12.57.34.636504	2	API	QIBM_QLNT_OPENDB_ALLO		* 00**/xenon/notes/data/homepage.nsf												
2001-03-01-12.57.34.658376	3	API	QIBM_QLNT_OPENDB_ALLO		* 00**names.nsf												
2001-03-01-12.57.34.673912	4	API	QIBM_QLNT_OPENDB_ALLO		* 00**/xenon/notes/data/names.nsf												
2001-03-01-12.57.34.685920	5	API	QIBM_QLNT_OPENDB_ALLO		* 00**names.nsf												
2001-03-01-12.57.34.687352	6	API	QIBM_QLNT_OPENDB_ALLO		* 00**/xenon/notes/data/names.nsf												
2001-03-01-12.57.34.699928	7	API	QIBM_QLNT_OPENDB_ALLO		* 00**/xenon/notes/data/homepage.nsf												
2001-03-01-12.57.34.790408	8	API	QIBM_QLNT_HTTP_URL	0	* 00**/homepage.nsf/homePage.gif?OpenImageResource												
2001-03-01-12.57.34.791552	9	API	QIBM_QLNT_OPENDB_ALLO		* 00**/xenon/notes/data/homepage.nsf												
2001-03-01-12.57.34.793664	10	API	QIBM_QLNT_HTTP_URL	0	* 00**/xenon/notes/data/domino/icons/ecblank.gif												
2001-03-01-12.57.34.826896	11	API	QIBM_QLNT_HTTP_URL	0	* 00**/homepage.nsf/smSpacer.gif?OpenImageResource												
2001-03-01-12.57.34.827976	12	API	QIBM_QLNT_OPENDB_ALLO		* 00**/xenon/notes/data/homepage.nsf												
2001-03-01-12.57.35.095536	13	API	QIBM_QLNT_OPENDB_ALLO		* 00**/xenon/notes/data/homepage.nsf												
2001-03-01-12.57.35.115440	14	API	QIBM_QLNT_OPENDB_ALLO		* 00**/xenon/notes/data/homepage.nsf												
2001-03-01-12.57.43.110456	15	API	QIBM_QLNT_OPENDB_ALLO		* 00**/xenon/notes/data/names.nsf												
2001-03-01-12.57.43.133040	16	API	QIBM_QLNT_OPENDB_ALLO		* 00**names.nsf												
2001-03-01-12.57.43.145712	17	API	QIBM_QLNT_OPENDB_ALLO		* 00**/xenon/notes/data/names.nsf												
2001-03-01-12.57.43.167616	18	API	QIBM_QLNT_OPENDB_ALLO		* 00**names.nsf												
					More...												
F3=Exit F12=Cancel F19=Left F20=Right F21=Split F22=Width 80																	
Cannot show this data at the display station.																	

Figure 4-50 Output from the SQL statement

Both analyses show you the job responsible for the event, the API (trace point) that was triggered, and the specific information associated with the event triggered. Whether it is a view, agent, or URL causing performance problems, these techniques will help you find patterns in the data and help isolate the problem.

4.14.6 TPROF PEX trace

In the event the system is not on a release of OS/400 or Domino that supports the Domino PEX trace points, Trace Profile (TPROF) PEX trace can be very useful in isolating CPU consumption problems. This trace is also beneficial to be used before enabling the Domino PEX trace points. It can uncover the problem or help pinpoint the level of Domino PEX trace points that need to be collected. TPROF is a PEX trace that helps identify system-wide CPU “hotspots” within application programs, service programs, OS/400 programs, procedures and modules, and SLIC procedures.

The PEX data is collected by using the Base PEX event PMCO. The data is formatted as if you had collected *PROFILE PEX data.

Running a TPROF PEX trace

To run a TPROF PEX trace, you need to perform the following steps:

1. Create a library for the data to be stored using the CRTLIB command:

```
CRTLIB LIB(MYLIB)
```

2. Add a PEX definition that describes the data to be collected:

```
ADDPEXDFN DFN(MYTRACE) TYPE(*TRACE) JOB(*ALL) TASK(*ALL) MAXSTG(100000) INTERVAL(5)  
TRCTYPE(*SLTEVT) SLTEVT(*YES) BASEVT(*PMCO)
```

3. Start the PEX trace:

```
STRPEX SSNID(MYTRACE) DFN(MYTRACE01)
```

4. Duplicate the performance problem.

5. End the PEX trace:

```
SBMJOB CMD(ENDPEX SSNID(MYTRACE01) DTALIB(MYLIB)) JOB(ENDPEX).
```

Note: Since most iSeries machines are server machines with limited amounts of interactive capabilities, the Submit Job command is used.

6. Once the job submitted in step 5 has ended, print the PEX data using PRTPEXRPT as shown below. This creates a spooled file under the PRTPEXRPT job.

```
SBMJOB CMD(PRTPEXRPT MBR(MYTRACE01) LIB(MYLIB) TYPE(*PROFILE) PROFILEOPT(*SAMPLECOUNT  
*PROCEDURE)) JOB(PRTPEXRPT)
```

Analyzing the TPROF trace

The output produced from step 6 in the previous section produces a report that shows which programs, modules, or procedures the system spent the most time in, based on hit counts. This is very useful in isolating a point in which a job or jobs are spending most of their time. Figure 4-51 provides an example of the TPROF PEX trace output.

Histogram	Hit Cnt	Hit %	Cum %	Start Addr	Map Flag	Stmt Numb	Name
**	15683	6.3	6.3	FFFFFFFFB31D04D4	++	0	HMFREHS/#hmfrehS
*	10323	4.1	10.4	FFFFFFFFB1618FE0	++	0	QUMUGA/pSpinWait__11QuMutexGateFv
*	<u>9667</u>	<u>3.9</u>	<u>14.3</u>	<u>30FAE17D640197E4</u>	==	0	<u>ACME DATALINK/CheckValidKey</u>
*	9205	3.7	18.0	FFFFFFFFB31E3140	++	0	HMALCHS/#hmalchs
*	7797	3.1	21.1	1E88ABC9373296D0	==	0	LIBNOTES OSSEM/OSLockSem
	6241	2.5	23.6	1E88ABC93733BDC0	==	0	LIBNOTES MEMLOCK/LockHandle
	6124	2.5	26.0	1E88ABC937329490	==	0	LIBNOTES OSSEM/OSUnlockSem
	4771	1.9	27.9	1E88ABC93732BC40	==	0	LIBNOTES OSSEM/OSUnlockSpin
	4387	1.8	29.7	1E88ABC937350A00	==	0	LIBNOTES THREAD/OSGetNativeThreadId
	3860	1.5	31.2	FFFFFFFFF800690	++	0	CFNSBLA/bla_NORTHSTAR
	3532	1.4	32.6	1E88ABC93732BE20	==	0	LIBNOTES OSSEM/StaticHangEnable
	3051	1.2	33.9	1E88ABC9373C5790	==	0	LIBNOTES VARRAY/OSAddressInVARRAY
	2971	1.2	35.1	FFFFFFFFF027030	++	0	STRNGEAO/memmovwteaofree
	2912	1.2	36.2	1E88ABC9375C84B4	==	0	LIBNOTES MD/MDTransform
	2611	1.0	37.3	1E88ABC93733EF00	==	0	LIBNOTES MEMORY/LockMemHandle
	2499	1.0	38.3	222A3D696B00ABE0	==	0	QPOWPTHREAD/thread_getthreadid_np
	2410	1.0	39.2	1E88ABC9376DD9A0	==	0	LIBNOTES DBUNK/DbUNKGetName
	2386	1.0	40.2	1E88ABC93733C490	==	0	LIBNOTES MEMLOCK/LockMem
	2225	0.9	41.1	FFFFFFFFF802830C0	++	0	PDCEACTV/_ct_14PdcEventActiveFCU1T1CQ _14PdcEventActive6CrtTestPC6TDTTask
	2189	0.9	42.0	00CA6A6DC4006190	==	0	QC2UTIL1 QC2ALLOC/malloc
	2074	0.8	42.8	30FAE17D640199F0	==	0	ACME DATALINK/GetValidName
	2073	0.8	43.6	1E88ABC937391500	==	0	LIBNOTES BPOOL/OSBB1ockAddr
	1916	0.8	44.4	FFFFFFFFF0030A0	++	0	CFSCV/scv0
	1823	0.7	45.1	1E88ABC93733B640	==	0	LIBNOTES MEMLOCK/OSUnlockObject
	1753	0.7	45.8	1E88ABC93733B7C0	==	0	LIBNOTES MEMLOCK/OSLockObject
	1748	0.7	46.5	1E88ABC937A13500	==	0	LIBNOTES ODS/ODSToOrFromHost
	1714	0.7	47.2	1E88ABC93733C330	==	0	LIBNOTES MEMLOCK/UnlockMem
	1650	0.7	47.9	80000000000030A0	++	0	HVSCVVEC/hvscvvec
	1641	0.7	48.5	FFFFFFFFFE6C2980	++	0	MASOUBLA/masounlockmutexbla
	1608	0.6	49.2	1E88ABC93733BD50	==	0	LIBNOTES MEMLOCK/UnlockHandle
	1573	0.6	49.8	1E88ABC937394040	==	0	LIBNOTES CMEM/Cmovmem
	1520	0.6	50.4	1E88ABC937325B80	==	0	LIBNOTES OSSEM/OSLockReadSem
	1484	0.6	51.0	FFFFFFFFFE6C26C0	++	0	MASOLBLA/masolockmutexbla

Figure 4-51 Sample page from a TPROF PEX report

In the example shown in Figure 4-51, note the user program entry that is underlined. By reviewing this program and the functions it performs within the application, you can improve the overall performance of the system. Most PEX collections are not always this informative, which may require more in-depth analysis by looking at the patterns within the data to see how it all fits together. Collecting PEX data with the Domino PEX trace points may be needed.



Domino logs and statistics

Domino includes a set of commands and statistical reports to monitor server performance. The *commands* provide an interactive method for checking performance statistics. The *statistical reports* provide a method of collecting the performance statistics on a scheduled interval. The statistical reports are stored in the default database – Statistics and Reporting Database (statrep.nsf).

Gathering statistics requires system resources. Keeping what you collect to a minimum improves the performance of your Domino server.

5.1 The commands

Of the numerous Domino console commands, the following five commands are probably the most useful when monitoring performance and investigating a performance problem:

- ▶ Show tasks
- ▶ Show stat
- ▶ Show database
- ▶ Show platform statistics
- ▶ Show trans

The commands can be run by using any of these methods:

- ▶ Enter the Work with Domino Servers (WRKDOMSVR) command on any OS/400 command line. Then, select the server, type option **8** (Work console), press Enter and enter the Domino console command **show xxxx**.
- ▶ Starting from a command line on the iSeries server, enter the Work With Domino Console (WRKDOMCLS) command. Then, select the **server**, and enter the Domino console command **show xxxx**.
- ▶ Start the Domino Administrator, and choose the **Server** tab and **Console**. Enter the command **show xxxx**.
- ▶ Run the Submit Domino Command (SBMDOMCMD) command from an OS/400 command line or from a CL program. Then, enter the Domino console command **show xxxx** and the server name.

In this case, you do not see the output of the command immediately. You can either redirect the output to a stream file for further analysis or see the results in the notes log LOG.NSF. To redirect the output of a Domino console command into a text file, append the greater than sign (>) followed by a file name to the command:

```
show xxxx >filename
```

The output file is stored in the Domino server directory within the iSeries Integrated File System (IFS). This is specially useful if you wrote a CL program that periodically writes a certain statistic value into a text file.

Domino commands and their parameters can be abbreviated as long as the remaining characters are not ambiguous. For example, the command **show tasks** can also be entered in the form **sh ta**.

5.1.1 The show tasks command

The **show tasks** Domino console command displays:

- ▶ The status of the active server tasks
- ▶ The Domino server release
- ▶ The name of the server
- ▶ The path of the Notes program directory
- ▶ Optionally the thread number of each task

Figure 5-1 shows an example of the output from running the **show tasks** console command.

```

Lotus Domino (r) Server (Release 5.0.8 for OS/400) 10/19/2001 09:44:48 AM

Server name:          applsvr/sg5162b
Server directory:     /domino/applsvr
Partition:           APPLSVR
Elapsed time:         00:27:18
Transactions/minute:  Last minute: 18; Last hour: 6; Peak: 39
Peak # of sessions:   2 at 10/19/2001 09:22:41 AM
Transactions:         202
Availability Index:   100 (state: AVAILABLE)
Message Tracking:     Not Enabled
Shared mail:         Not Enabled
Number of Mailboxes:  1
Pending mail: 0       Dead mail: 0
Waiting Tasks:       0
Transactional Logging: Not Enabled

      Task                Description
-----
Database Server          Perform console commands
Database Server          Listen for connect requests on TCPIP
Database Server          Load Monitor is idle
Database Server          Database Directory Manager Cache Refresher is idle
Database Server          Idle task
Database Server          Idle task
Database Server          Perform Database Cache maintenance
Database Server          Idle task
Database Server          Idle task
Database Server          Server for Notes Admin/sg5162b on TCPIP
Database Server          Server for Donald Duck/sg5162b on TCPIP
Admin Process            Idle
Event Monitor            Idle
Calendar Connector       Idle
Schedule Manager         Idle
Admin Process            Idle
Agent Manager            Executive '1': Idle
Agent Manager            Idle
Stats                    Idle
Indexer                  Idle
Router                   Idle
Replicator               Idle
HTTP Web Server          Listening on port(s) 8081
DECS Server              Idle

```

Figure 5-1 The **show tasks** output

An undocumented debug parameter provides the iSeries server job and thread number associated with tasks running under the SERVER job. The debug parameter helps map a thread to a user or a system function

Figure 5-2 shows an example of the output from running the **show task debug** console command.

Lotus Domino (r) Server (Release 5.0.8 for OS/400) 11/05/2001 08:08:06 AM

Server name: applsvr/sg5162b
 Server directory: /domino/applsvr
 Partition: APPLSVR
 Elapsed time: 00:03:30
 Transactions/minute: Last minute: 1; Last hour: 5; Peak: 9
 Peak # of sessions: 2 at 11/05/2001 08:06:19 AM
 Transactions: 19
 Availability Index: 100 (state: AVAILABLE)
 Message Tracking: Not Enabled
 Shared mail: Not Enabled
 Number of Mailboxes: 1
 Pending mail: 0 Dead mail: 0
 Waiting Tasks: 0
 Thread Pools: Enabled
 Transactional Logging: Enabled

Task	Description
17853:0012: 0003 Database Server	Perform console commands
17853:0013: 0004 Database Server	Listen for connect requests on TCPIP
17853:0014: 0005 Database Server	Load Monitor is idle
17853:0015: 0006 Database Server	Database Directory Manager Cache Refresher is idle
17853:0016: 0007 Database Server	Thread pool member on TCPIP
..	
17853:003D: 002E Database Server	Thread pool member on TCPIP
17853:003E: 002F Database Server	Thread pool utility task on TCPIP
17853:003F: 0031 Database Server	Idle task
17853:0040: 0032 Database Server	Idle task
17853:0041: 0033 Database Server	Perform Database Cache maintenance
17853:0042: 0034 Database Server	Idle task
..	
17853:0049: 003B Database Server	Idle task
17853:004A: 003C Database Server	Recovery Manager Checkpoint Thread
17853:004B: 003D Database Server	Recovery Manager DB Flushing Thread
17853:0000: 003E Database Server	Server for Notes Admin/sg5162b on TCPIP
Event Monitor	Idle
Calendar Connector	Idle
Schedule Manager	Idle
Admin Process	Idle
Agent Manager	Executive '1': Idle
Agent Manager	Idle
Stats	Idle
Indexer	Idle
Router	Idle
Replicator	Idle
HTTP Web Server	Listening on port(s) 8081
DECS Server	Idle

Figure 5-2 The show tasks debug results

In the Task column, there is a server task with a thread identified as 17853:0000: 003E. On the iSeries server, the thread number is eight digits; only the last four digits are shown here. In this example, you can see that the thread is used by Notes Admin/sg5162b, so you know who to talk to if there is a performance problem or if a function is not being used properly.

To analyze a performance problem, display the call stack for the thread you suspect that has a performance issue. To find out what job 17853:0000 003E is doing, perform the following steps:

1. Use the Work with Domino Servers (WRKDOMSVR) command to display a list of all Domino servers.
2. Select option **9** (Work server jobs) next to the server name.
3. Select option **12** (Work with threads) next to the job name.
4. Select option **10** (Display call stack) next to the thread 0000003E.

You see a panel similar to the Display Call Stack example shown in Figure 5-3.

Display Call Stack				
			System:	AS06
Job:	SERVER	User:	QNOTES	Number: 017853
Thread: 0000003E				
Program				
Rqs	or			
Lvl	Procedure	Library	Statement	Instruction
	< tth_parm_t	QSYS	0000001092	
	< eate_part2	QSYS	0000001587	
	< eadWrapper	QNOTES	0000000008	
	Scheduler	QNOTES	0000000032	
	< tilityTask	QNOTES	0000000024	
	< elayThread	QNOTES	0000000001	
	< nix_usleep	QNOTES	0000000006	
	select	QSYS	0000004667	
Bottom				
F3=Exit	F10=Update stack	F11=Display activation group	F12=Cancel	
F16=Job menu	F17=Top	F18=Bottom	F22=Display entire name	

Figure 5-3 Call stack for thread 0000003E

Tip: In 7.12, “Excessive translations on iSeries and zSeries” on page 202, you can see another example that explains how to display the call stack and shows a problem with excessive CCSID translations.

Display Call Stack: Panel group update to improve readability

If you have Domino 5.03 or higher installed, you may improve the analysis of the call stack for Domino server tasks. A new panel group named GWVJOB is provided in library QNOTES. This panel group enables a more readable display of the Call Stack.

To replace the panel group QGWVJOB in QSYS with the panel group shipped with the Lotus Domino product, enter the CL commands in the order shown below on any OS/400 command line:

```
CRTLIB GWVLIB TEXT('Panel Group for better Domino Call Stack display')
CRTDUPOBJ GWVJOB QNOTES *PNLGRP GWVLIB QGWVJOB
CHGSYSLIBL GWVLIB
```

Figure 5-4 shows the Call Stack view with improved readability.

Display Call Stack			
Job: SERVER	User: QNOTES	Number: 017839	System: AS06
Thread: 0000003E			
Procedure	Statement	ILE Module	ILE program
LE_Create_Thread2__FP12crtth_parm_t	0000001092	QLECRTTH	QLESPI
pthread_create_part2	0000001587	QPOWSPTHR	QPOWPINT
ThreadWrapper	0000000008	THREAD	LIBNOTES
Scheduler	0000000035	SCHED	SERVER
OSDelayThread	0000000001	THREAD	LIBNOTES
unix_usleep	0000000006	USLEEP	LIBNOTES
select	0000004667	QPOLLIB1	QPOLLIB1
Bottom			
F3=Exit	F10=Update stack	F11=Display instruction	F12=Cancel
F16=Job menu	F17=Top	F18=Bottom	F22=Display entire name

Figure 5-4 Call stack view with improved readability

Important: The panel group that is used for this function only works well with Domino jobs. If you attempt to display the call stacks of non-Domino jobs while using this panel group, you will receive incomplete information. To correct this problem, you must sign off and sign back on or run the command:

```
CHGSYSLIBL LIB(GWVLIB) OPTION(*REMOVE)
```

5.1.2 Understanding the Domino statistics

The Domino console command **show stat** shows server statistics that relate to:

- ▶ The number of transactions
- ▶ The number of concurrent users
- ▶ The available amount of disk space
- ▶ The memory available and used
- ▶ Mail, replication, and network activity
- ▶ Other useful information

You can use this command without any parameters to show all available statistics, or select a group or a single value by specifying its name. To display only a group of statistics, specify the group name (the characters on the left of the period) followed by an asterisk (*) as a wildcard.

You can find a complete list of statistic descriptions in the Domino events4.nsf database, in the Names & Messages, Statistics Names view.

For monitoring performance and investigating a performance problem, we recommend that you run the following commands:

- show stat Server.Trans.PerMinute
- show stat Server.Users
- show stat Server.Sessions.Dropped
- show stat Database.BufferPool*
- show stat Domino
- show database

Note: To reset a statistic that is cumulative, you can use **Set Statistics statisticname**. *Statisticname* is a required parameter that names the statistic to be reset. You can't use wildcards (*) with this argument.

For example, **Set Stat Server.Trans.Total** resets the **Server.Trans.Total** statistic to 0.

Domino transactions per minute

The **show stat Server.Trans.PerMinute** statistic provides the number of transactions that occurred on the server during the last minute. This information is useful for monitoring overall server use.

Figure 5-5 shows an example of the output from running the **show stat Server.Trans.PerMinute** console command.

```
Server.Trans.PerMinute = 2185
```

Figure 5-5 The **show stat Server.Trans.PerMinute** statistic

Number of active Notes users

The **show stat** console command can also display a group of statistic values by specifying the wildcard character **"*"**. For example, **show stat Server.Users*** provides the current number of active Notes sessions as well as the peaks and the times when they appeared. It includes all users and all server tasks connecting to the server (for example, the Replicator). It shows information about peak times and is useful for monitoring overall server use.

The example in Figure 5-6 shows three users that are currently active on the server. On March 13 at 11:42 AM, the most users (65) were active on the server (since the server was started or the statistics were reset). On March 14 at 02:01 PM, the most users (17) performed any network operation to this server in one minute. And on March 14 at 09:14 AM, the most users (26) performed any network operation to this server within a five minute time frame.

```
Server.Users = 3
Server.Users.1MinPeak = 17
Server.Users.1MinPeakTime = 03/14 02:01:55 PM
Server.Users.5MinPeak = 26
Server.Users.5MinPeakTime = 03/14 09:14:03 AM
Server.Users.Peak = 65
Server.Users.PeakTime = 03/13 11:42:02 AM
```

Figure 5-6 The **show stat Server.Users*** statistic

The parameters shown in Figure 5-6 are explained here:

- **Server.Users:** This specifies the number of active Notes Sessions. This includes all users and all server tasks connecting to the server (for example, the Replicator). In the example shown in Figure 5-6, three users are currently connected to the server.

- ▶ **Server.Users.1MinPeak:** This specifies the number of users who have, in the last minute, performed a Notes transaction (some network activity caused, for example, when opening a database or document on the server). This value *does not* specify the number of users or transactions in the last minute. The Domino server checks this data regularly and updates the value only if the number is greater than the previous value.
- ▶ **Server.Users.1MinPeakTime:** This value shows the date/time at which the maximum number of users have done a network operation in the past minute. Note that Server.Users.1MinPeak specifies the *number* of users during this past minute. Users who have not done any transaction so far, but are waiting for a transaction, are also counted in this list.
- ▶ **Server.Users.5MinPeak** and **Server.Users.5MinPeakTime:** This shows the number of users and the time, respectively, at which the server had the most users performing a transaction during the past five minutes.
- ▶ **Server.Users.Peak** and **Server.Users.PeakTime:** This shows the maximum number of users and the time, respectively, for users who had an active connection to the server.

Sessions dropped

The `show stat Server.Sessions.Dropped` statistic provides the number of users who gave up trying to connect to the server (usually by restarting or pressing Ctrl-Break). This statistic is useful for monitoring server response time.

The `Server.Sessions.Dropped` statistic indicates poor server performance if the number of dropped sessions is high on a regular basis. Resolving a problem with dropped sessions may require:

- ▶ Cleaning up and reducing the size of databases
- ▶ Moving databases and users to a different server
- ▶ Adding a server
- ▶ Adding memory
- ▶ Adding DASD
- ▶ Upgrading the processor

Figure 5-7 shows an example of the output from running the `show stat Server.Sessions.Dropped` server command.

```
Server.Sessions.Dropped = 0
```

Figure 5-7 The `show stat Server.Sessions.Dropped` command

Domino database buffer pool statistics

Monitor how much NSF buffer pool your server uses by running the `show stat database` command. If `BufferPoolPeak` is routinely more than 95% of the maximum, you may need to increase the `NSF_Buffer_POOL_Size`. For more information about the `NSF_Buffer_POOL_Size`, see 7.14.1, “`NSF_Buffer_Pool_Size` or `Nsf_Buffer_Pool_Size_mb`” on page 211.

Also monitor `Database.BufferPool.PerCentReadsInBuffer`. In our tests under heavy workload, we saw a `BufferPoolPeak` of 100%, while `PerCentReadsInBuffer` was under 95%. Under a light load, `BufferPoolPeak` was under 10%, while `PerCentReadsInBuffer` was over 97%.

When the maximum size of NSFBufferPool is reached, old entries are taken out of the pool to make room for the new entries. At this point, the Database.BufferPool.PercentReadsInBuffer statistic will begin to decrease. This is not good because a low number in this pool indicates that there is not enough memory to cache most of the reads. The higher the percentage is, the better. This statistic represents the cumulative percentage of time that the buffer pool was used to service requests.

The Domino server **show stat database.Database.BufferPool*** command presents a screen like the example shown in Figure 5-8.

```
Database.Database.BufferPool.Maximum.Megabytes = 300
Database.Database.BufferPool.MM.Reads = 0
Database.Database.BufferPool.MM.Writes = 0
Database.Database.BufferPool.Peak.Megabytes = 3
Database.Database.BufferPool.PercentReadsInBuffer = 97.28
```

Figure 5-8 The **show stat database.Database.BufferPool*** command

The parameters shown in Figure 5-8 are explained here:

► **Database.BufferPool.Maximum.Megabytes**

This is the maximum allowed in the NSF Buffer Pool. It defaults to one-fourth (¼) of available physical RAM. If the notes.ini parameter NSF_BUFFER_POOL_SIZE=nnnnnn is set, this parameter overrides the default.

► **Database.BufferPool.MM.Reads**

This is the cumulative number of times the buffer has been read.

► **Database.BufferPool.MM.Writes**

This is the cumulative number of times that data has been written to the buffer.

► **Database.BufferPool.Peak.megabytes**

This is the largest that the pool has grown to, so far. This value will never decrease until the Notes server is restarted.

► **Database.BufferPool.PercentReadsInBuffer**

This is the cumulative percentage of time that the buffer pool was used to service requests. The higher this number is, the better.

HTTP server statistics

The **show stat domino** command monitors HTTP server activities on the Domino server. The Domino **show stat Domino.Config.ActiveThreads*** console command shows the server document settings (Figure 5-9) for Number active threads and Minimum active threads.

```
Domino.Config.ActiveThreads.Max = 40
Domino.Config.ActiveThreads.Min = 20
```

Figure 5-9 The **show stat Domino.Config.ActiveThreads*** command

The parameters in Figure 5-9 are explained here:

► **Domino.Config.ActiveThreads.Max**: This is the maximum number of active threads servicing HTTP requests.

► **Domino.Config.ActiveThreads.Min**: This is the minimum number of active threads servicing HTTP requests.

The Domino **show stat Domino.Threads*** console command shows the peak number of active threads.

- ▶ **Domino.Threads.Active.Peak:** This indicates the peak number of active threads since Domino was started.
- ▶ **Domino.Threads.Peak.Total:** This value represents the total peak number of active threads since Domino was started.
- ▶ **Domino.Threads.Peak.Time:** This value indicates the time during the peak of the total thread statistics from above. Domino.Config.ActiveThreads.Max should be greater than Domino.Theads.Active.Peak.
- ▶ **Domino.Config*:** If you run the **show stat domino.config*** command, you will see a list of how HTTP server is configured. This list may be useful if you don't have access to the server document in the Domino directory. Figure 5-10 shows an example of the output from Domino **show stat domino.config*** console command.

```
Domino.Config.ActiveThreads.Max = 40
Domino.Config.ActiveThreads.Min = 20
Domino.Config.AllowDirectoryLinks = 1
Domino.Config.Directory.CGI = /domino/applsrvr/domino/cgi-bin
Domino.Config.Directory.HTML = /domino/applsrvr/domino/html
Domino.Config.Directory.Icons = /domino/applsrvr/domino/icons
Domino.Config.Directory.JavaRoot = /domino/applsrvr/domino/java
Domino.Config.DNSLookup = 0
Domino.Config.HomeURL = /homepage.nsf?Open
Domino.Config.HostName =
Domino.Config.Image.Format = GIF
Domino.Config.Image.Interlaced = 1
Domino.Config.Log.Access = access
Domino.Config.Log.Error = error
Domino.Config.Log.Filter =
Domino.Config.Log.TimeStamp = 0
Domino.Config.PortNumber = 8081
Domino.Config.PortStatus = 1
Domino.Config.SSL.KeyFile = /domino/applsrvr/keyfile.kyr
Domino.Config.SSL.PortNumber = 443
Domino.Config.SSL.Status = 0
Domino.Config.Timeout.CGI = 5
Domino.Config.Timeout.IdleThread = 0
Domino.Config.Timeout.Input = 2
Domino.Config.Timeout.Output = 20
Domino.Config.URLpath.CGI = /cgi-bin
Domino.Config.URLpath.Icons = /icons
Domino.Config.URLpath.JavaRoot = /domjava
Domino.Config.URLpath.Servlet = /servlet
Domino.Config.View.Lines = 30
Domino.Config.WelcomePage = default.htm
```

Figure 5-10 The **show stat domino.config*** command

5.1.3 Show Domino open database statistics (sh DBS)

The Show Open Databases (**show DBS**) command is a tool for monitoring the performance of open databases. This command returns the information shown in Figure 5-11.

Database Name	Refs	Mod	FDs	LockWaits/AvgWait	#Waiters	MaxWaiters
/domino/applsrvr/statrep.nsf	1	Y	1	0	0	0
/domino/applsrvr/events4.nsf	8	N	1	0	0	0
/domino/applsrvr/busytime.nsf	1	N	1	0	0	0
/domino/applsrvr/statmail.nsf	1	N	1	0	0	0
/domino/applsrvr/mail.box	1	N	1	0	0	0
/domino/applsrvr/decsadm.nsf	3	N	1	0	0	0
/domino/applsrvr/names.nsf	55	Y	9	0	0	1
/domino/applsrvr/log.nsf	1	Y	1	0	0	0

Figure 5-11 The show database command

The columns shown in Figure 5-11 are explained here:

- **Refs:** The number of times the database has been opened (the DBHANDLE count for the database).
- **Mod:** Whether the database has been modified, but not yet flushed to disk.
- **FDs:** The number of file descriptors currently being used for the database.
- **LockWaits:** The number of times a user has had to wait for a lock on the database (read or write).
- **AvgWait:** The average wait time in milliseconds for each wait.
- **#Waiters:** The number of waiters currently on the database lock. (This number changes rapidly.)
- **MaxWaiters:** The maximum number of waiters ever on the database lock.

Note: To display LockWaits and AvgWait values, you must temporarily add the setting COLLECT_DB_LOCK_WAITS=1 to your server's notes.ini file. This setting consumes server resources, so you should remove it after you view Show DBS statistics.

5.1.4 Platform-dependent statistics

New with Domino 5.0.3 for iSeries is the integration of platform and Domino statistics. Platform statistics are enabled by adding PLATFORM_STATISTICS_ENABLED=1 to the notes.ini file and restarting your server. This feature is disabled by default.

After you enable the platform-dependent statistics in notes.ini, you can only see them on the console through the **show stat platform** command. The next step is to make sure those statistics are also recorded in the statrep.nsf database so you can analyze them.

To collect performance data from Domino, you need to configure a couple of things as explained in the following process:

1. From your Collection Server (usually a hub in a large environment), open the **Statistics & Events database** (events4.nsf). This database allows you to set up the collection interval of your Domino servers.
2. Click **Server Statistic Collection** on the left. This displays a Collection Document. Edit this document or create a new one if one does not exist.

Figure 5-12 shows the Statistics & Events database views.

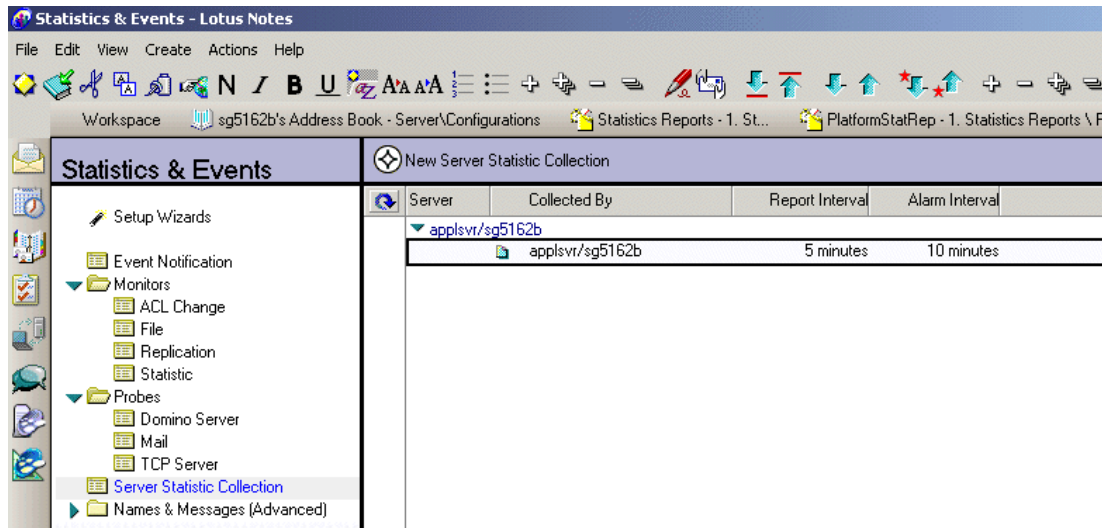


Figure 5-12 Statistics & Events database views

- Once the Server Statistic Collection document is opened for editing, click the **Basics** tab as shown in Figure 5-13. Be sure to enter the server name of the server that will collect the statistics. Select the option to collect statistics from **All servers in this domain** or specify the names of the servers from which you want to collect performance data. Once this is complete, click the **Options** tab.

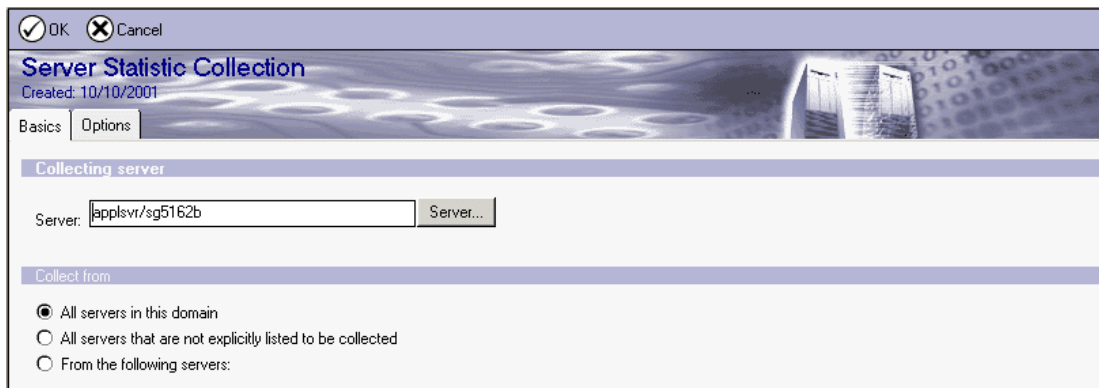


Figure 5-13 Server Statistic Collection document

- Under the Options tab, click the **Log statistics a database** check box and specify the database that will gather the statistics as shown in Figure 5-14. We recommend you collect the statistics with the statrep.nsf database, and for performance purposes, collect statistics every 15 minutes with alarms every 60 minutes.

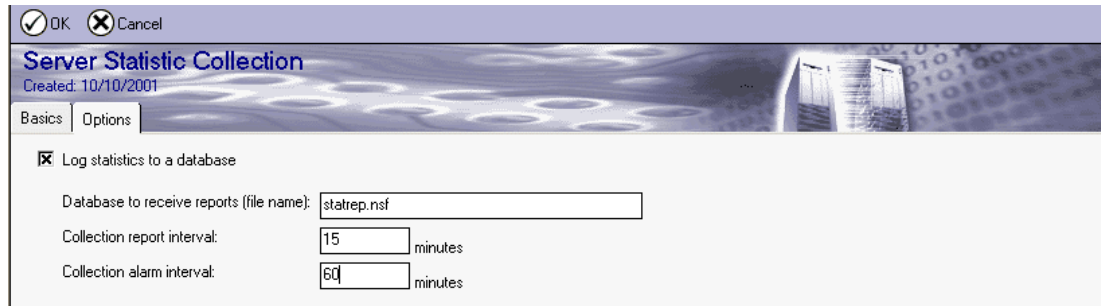


Figure 5-14 Server Statistic Collection document options

5. Edit the notes.ini file of the server collecting the statistics by adding the line `PLATFORM_STATISTICS_ENABLED=1`. Figure 5-18 on page 149 shows an example of how to edit the notes.ini file using Operations Navigator. You can also use option 13 in the Work with Domino Servers (WRKDOMSVR) panel to edit the notes.ini file.

This ensures the Domino servers gathers platform statistics unique to iSeries, like total CPU utilization, number of page faults per second, number of Wait-to-Ineligible transitions for threads in the memory pool, and CPU utilization per Domino tasks, which is very important for performance analysis.

6. After you save the notes.ini file, replicate your events4.nsf database to all servers in your domain by issuing the command:

```
push <server name> events4.nsf
```

Here, *server name* is the destination server. Make sure to specify a fully qualified server name. Otherwise, you see the message “Access control is set in events4 to not allow replication from <server name> events4.nsf.”

7. After all servers receive the updated file, recycle your servers (end and restart). When they restart, they will begin collecting their Domino statistics in the statrep.nsf file of your destination server.

Metrics are collected continuously, unless they are disabled or reset, or the server is stopped. The server resets this information each time it is started. OS/400 platform statistics include performance metrics for CPU, memory, disk, and process utilization.

The Statistics and Reports database (statrep.nsf) includes the platform view, Show Platform Statistics. However, this does not include iSeries-specific data, except the overall CPU utilization. If you want to see other iSeries related statistics, you need to create an additional view. See 5.3, “Combining performance data from Domino and iSeries” on page 152.

The Statistics and Reports database is further described in 5.2.2, “The Statistics and Reporting database (statrep.nsf)” on page 149.

System statistics

System statistics (Platform.System.TotalUtil.*) indicate the platform system total utilization. It is the current average CPU utilization for the system. The minimum, average, and maximum utilization for each of the system statistics values also available.

System statistics collect the same information for each partition, and the same information is reported for each Domino server. The values reported in each server may vary due to timing of the collection.

Total CPU utilization

The total CPU utilization includes the entire workload of the iSeries server, that is all Domino servers and all non-Domino jobs. For the CPU utilization of specific Domino tasks of a single server, see “Process statistics” on page 140. The average CPU percent utilization of all processors on the system is reported by the value of *Platform.System.TotalUtil*.

The smallest TotalUtil sampling for a session is contained in *Platform.System.TotalUtil.Min*.

The average of all TotalUtil samplings for a session can be found in *Platform.System.TotalUtil.Avg*. This is calculated by the finding the sum of all TotalUtil samplings divided by the number of samplings.

The largest TotalUtil sampling for a session is reported with *Platform.System.TotalUtil.Max*.

Memory statistics

Memory statistics (*Platform.Memory.**) are collected and reported for the storage pool in which the Domino server is running. If more than one server (partition) shares the storage pool, the memory statistics reported include all servers. The values reported in each server may vary due to timing of the collection.

The *Platform.memory** statistics include:

- ▶ The amount of memory in kilobytes allocated to the pool that this Domino partition is running in.
 - *Platform.Memory.KBSize*
- ▶ The number of page faults per second in the memory pool this server is using.
 - *Platform.Memory.FaultsPerSec*
 - *Platform.Memory.FaultsPerSec.Min*
 - *Platform.Memory.FaultsPerSec.Avg*
 - *Platform.Memory.FaultsPerSec.Max*
- ▶ The number of pages per second read to or written from the disk with the memory pool this server is using.
 - *Platform.Memory.PagesPerSec*
 - *Platform.Memory.PagesPerSec.min*
 - *Platform.Memory.PagesPerSec.avg*
 - *Platform.Memory.PagesPerSec.max*
- ▶ The number of Wait-to-Ineligible transitions for threads in the memory pool this server is using. Any value greater than zero indicates your max active in the memory pool is too low.
 - *Platform.Memory.WaitToIneligible*

Process statistics

Process statistics (*Platform.Process.**) are independent from partition to partition; no values are shared. Some of the server tasks, such as AMGR, REPLICA, SERVER, and UPDATE, can have statistics collected and reported for up to four instances of the process. If an instance of a process is not present, 0 is reported. Any new process takes two collections before it is reported by statistics. This applies for processes started when the server is started or processes started later with the **load** command.

If a process is removed, the metric for that process goes to 0. If a new process begins, it uses the first unused statistic slot. For example, if there are three AMGR processes, process 2 ends, so you only receive statistics from Platform.Process.AMGR.1.Util and Platform.Process.AMGR.3.Util. If another AMGR starts, it reports its statistics to the first unused statistic slot, which is Platform.Process.AMGR.2.Util.

The *Platform.process** statistics include:

- ▶ The CPU utilization of the first ADMINP, AMGR, HTTP, REPLICATION, ROUTER, SERVER, and UPDATE process of this partition. This is the percentage of the sample interval that the process used the CPU.
 - Platform.Process.ADMINP.1.Util
 - Platform.Process.AMGR.1.Util
 - Platform.Process.HTTP.1.Util
 - Platform.Process.REPLICATION.1.Util
 - Platform.Process.ROUTER.1.Util
 - Platform.Process.SERVER.1.Util
 - Platform.Process.UPDATE.1.Util
- ▶ The CPU utilization of the second AMGR, REPLICATION, SERVER, and UPDATE process of this partition. This is the percentage of the sample interval that the process used the CPU.
 - Platform.Process.AMGR.2.Util
 - Platform.Process.REPLICATION.2.Util
 - Platform.Process.SERVER.2.Util
 - Platform.Process.UPDATE.2.Util
- ▶ The CPU utilization of the third AMGR, REPLICATION, SERVER, and UPDATE of this partition. This is the percentage of the sample interval that the process used the CPU.
 - Platform.Process.AMGR.3.Util
 - Platform.Process.REPLICATION.3.Util
 - Platform.Process.SERVER.3.Util
 - Platform.Process.UPDATE.3.Util
- ▶ The CPU utilization of the fourth AMGR, REPLICATION, SERVER, and UPDATE process of this partition. This is the percentage of the sample interval that the process used the CPU.
 - Platform.Process.AMGR.4.Util
 - Platform.Process.REPLICATION.4.Util
 - Platform.Process.SERVER.4.Util
 - Platform.Process.UPDATE.4.Util

Disk statistics

Disk statistics (Platform.LogicalDisk.*) are collected and reported for the whole system. All configured disks in all ASPs are collected. The Percent Used statistic is the percent of all space of all configured disks that are in use. The Percent Busy statistic is an average of the percent busy time for every configured disk.

The *Platform.LogicalDisk** statistics include:

- ▶ The average of the percent time that all configured disks in all ASPs are busy. The first “_Total” refers to all physical disks, and the second “_Total” refers to all logical disks. For the iSeries, read it as the total of all disks configured.
 - Platform.LogicalDisk._Total.1._Total.1.PctTime = .1
 - Platform.LogicalDisk._Total.1._Total.1.PctTime.avg = 1.4
 - Platform.LogicalDisk._Total.1._Total.1.PctTime.max = 45.7
 - Platform.LogicalDisk._Total.1._Total.1.PctTime.min = 0

- The percentage of all configured disks in all ASPs that is currently allocated for data.
 - Platform.LogicalDisk._Total.1._Total.1.PctUsed = 15.6

Figure 5-15 shows an example of the output from running the **show stat platform** console command.

```
Platform.LogicalDisk._Total.1._Total.1.PctTime = .1
Platform.LogicalDisk._Total.1._Total.1.PctTime.avg = 1.4
Platform.LogicalDisk._Total.1._Total.1.PctTime.max = 45.7
Platform.LogicalDisk._Total.1._Total.1.PctTime.min = 0
Platform.LogicalDisk._Total.1._Total.1.PctUsed = 15.6
Platform.Memory.FaultsPerSec = .3
Platform.Memory.FaultsPerSec.avg = .4
Platform.Memory.FaultsPerSec.max = 8.1
Platform.Memory.FaultsPerSec.min = 0
Platform.Memory.KBSize = 3,600,740
Platform.Memory.PagesPerSec = .3
Platform.Memory.PagesPerSec.avg = 2.1
Platform.Memory.PagesPerSec.max = 63.8
Platform.Memory.PagesPerSec.min = 0
Platform.Memory.WaitToIneligible = 0
Platform.Process.ADMINP.1.Util = 0
Platform.Process.AMGR.1.Util = 0
Platform.Process.AMGR.2.Util = 0
Platform.Process.AMGR.3.Util = 0
Platform.Process.AMGR.4.Util = 0
Platform.Process.HTTP.1.Util = 0
Platform.Process.REPLICA.1.Util = 0
Platform.Process.REPLICA.2.Util = 0
Platform.Process.REPLICA.3.Util = 0
Platform.Process.REPLICA.4.Util = 0
Platform.Process.ROUTER.1.Util = 0
Platform.Process.SERVER.1.Util = .8
Platform.Process.SERVER.2.Util = 0
Platform.Process.SERVER.3.Util = 0
Platform.Process.SERVER.4.Util = 0
Platform.Process.UPDATE.1.Util = 0
Platform.Process.UPDATE.2.Util = 0
Platform.Process.UPDATE.3.Util = 0
Platform.Process.UPDATE.4.Util = 0
Platform.System.TotalUtil = 2.2
Platform.System.TotalUtil.avg = 1.6
Platform.System.TotalUtil.max = 8.5
Platform.System.TotalUtil.min = 0
```

Figure 5-15 The show stat platform results

The Domino console platform command

Platform statistics collection is controlled by the PLATFORM command. The following keywords can be used with the PLATFORM command:

- **Time:** Display or change the current collection interval
- **Reset:** Reset collection of the platform statistics
- **Wait:** Stop collection of platform statistics
- **Run:** Start collection of platform statistics

For example, to start a new performance data monitoring session with a sampling rate of five minutes, type the following commands at the console:

1. `platform time 5`

The server collects performance data every five minutes.

2. `platform reset`

Statistic values sampled before this command was issued are not used in calculating the average, maximum, or minimum.

5.1.5 Show Domino Transactions

The **show transactions**, or in its short form **sh tran**, Domino console command provides a performance statistics summary for each Domino kernel transaction that is run. For each type of transaction, it displays the total number of NRPC transactions, the minimum and maximum duration of the transaction, the total time to perform all transactions, and the average time to perform the transaction. Figure 5-16 shows an example of the output from running the **show trans** console command.

Lotus Domino (r) Server (Release 5.0.8 for OS/400) 10/19/2001 01:59:06 PM

Server name: applsvr/sg5162b
 Server directory: /domino/applsvr
 Partition: APPLSVR
 Elapsed time: 04:41:37
 Transactions/minute: Last minute: 0; Last hour: 0; Peak: 50
 Peak # of sessions: 3 at 10/19/2001 09:47:40 AM
 Transactions: 294
 Availability Index: 100 (state: AVAILABLE)
 Message Tracking: Not Enabled
 Shared mail: Not Enabled
 Number of Mailboxes: 1
 Pending mail: 0 Dead mail: 0
 Waiting Tasks: 0
 Thread Pools: Enabled
 Transactional Logging: Not Enabled

Function	Count	Min	Max	Total	Average
OPEN_DB	14	1	21	63	4
GET_SPECIAL_NOTE_ID	6	0	1	2	0
OPEN_NOTE	105	1	269	1734	16
UPDATE_NOTE	4	7	31	61	15
DB_INFO_GET	2	1	1	2	1
DB_MODIFIED_TIME	8	0	1	7	0
DB_REPLINFO_GET	4	0	1	1	0
CLOSE_DB	14	0	9	16	1
CLOSE_COLLECTION	21	0	1	11	0
OPEN_COLLECTION	21	3	361	1387	66
UPDATE_FILTERS	2	0	1	1	0
READ_ENTRIES	22	1	126	424	19
LOCATE_NOTE	1	0	0	0	0
FIND_BY_KEY	1	1	1	1	1
NIF_OPEN_NOTE	4	3	4	14	3
NAME_LOOKUP	4	5	11	26	6
NAME_GET_AB	1	2	2	2	2
#81	1	5	5	5	5
GET_NAMED_OBJECT_ID	12	0	1	9	0
POLL_DEL_SEQNUM	15	1	2	27	1
GET_MULT_NOTE_INFO_BY_UNI	1	1	1	1	1
SERVER_TIME	1	1	1	1	1
DB_QUOTA_GET	2	1	2	3	1
FOLDER_GETIDTABLE	4	1	4	11	2
START_SERVER	11	61	701	2033	184
GET_UNREAD_NOTE_TABLE	9	0	1	8	0
GET_DBOPTIONS	2	0	0	0	0
FINDDESIGN_NOTES	1	0	0	0	0
GET_DBINFOFLAGS	1	0	0	0	0

Figure 5-16 The show trans output

Each column is explained here:

- **Function:** The kernel transactions provided by Domino.
- **Count:** The number of times the function was run.
- **Min:** The minimum CPU time in milliseconds it took to run the function.
- **Max:** The maximum CPU time in milliseconds it took to run the function.
- **Total:** The total CPU time in milliseconds spent running the function.
- **Average:** Indicates the total per count.

Tracking these statistics in a graph helps to determine if transactions require more or less CPU time as your use of the server grows.

The average of all transactions in CPU milliseconds may be useful for identifying a Queuing Multiplier:

- ▶ A *good response* is less than (<) 1000 milliseconds.
- ▶ An *average response* is greater than (>) 1000 milliseconds and less than (<) 5000 milliseconds.
- ▶ A *poor response* is greater than (>) 5000 milliseconds.

See 5.4, “Monitoring the Notes Remote Procedure Calls (NRPC)” on page 156, for a description how to analyze the transactions generated by a single Notes client.

Note: For Internet workloads, use the **show stat** command to monitor statistics. For example, enter these commands at the server console:

- ▶ SH STAT SMTP
- ▶ SH STAT POP3
- ▶ SH STAT IMAP
- ▶ SH STAT LDAP
- ▶ SH STAT Domino (for HTTP server statistics)

5.2 The reports

The Domino server provides services and tasks that create and report information about the Domino system. This information comes in two forms: statistics and events. *Statistics* show the status of processes running on the system. *Events* are generated when something takes place on the system. For example, the event “Replicating files with servername” occurs when a file replicates with a specified server.

Domino collects and stores statistics in two databases:

- ▶ **Notes log database (log.nsf)** is created automatically when you start a server for the first time. The Notes log database contains server events, such as replications performed, mail routed, and databases used. In addition to standard information that is logged for a server, you can customize what is logged for a server. For monitoring performance, customization is discussed in “Controlling what is logged in the database” on page 146.
- ▶ **Statistics and Reporting Database (statrep.nsf)** is created automatically when the server tasks Event or Collect is started for the first time. The Statistics and Reporting Database (statrep.nsf) contains the statistical reports that are created based on the server tasks running on the Domino server. For example, if the HTTP task is *not* running, no statistics related to the Web server are recorded.

We describe the capabilities of statrep.nsf in 5.2.2, “The Statistics and Reporting database (statrep.nsf)” on page 149. See also 5.3, “Combining performance data from Domino and iSeries” on page 152.

Combining statistic into a graph with iSeries server performance monitor data (such as CPU utilization, pool size, database/non-database faults, average disk response time per I/O operation, and average line utilization) can provide a method for tracking performance based on server usage and available system resources.

5.2.1 The Log database (log.nsf)

Every Domino server has a Log database that reports all server activity and provides detailed information about databases and users on the server. The terms *Notes Log database* and *Log File* are synonymous.

The Notes Log database is more for server maintenance activities versus monitoring performance or investigating performance issues. Check the Notes Log database for problems that relate to replication, mail routing, modem communications, and other scheduled server tasks.

Setting up and maintaining the Notes log database is handled by the notes.ini setting log:

Log Syntax: Log=logfilename, log_option, not_used, days, size

This syntax controls the name of the log file, whether events are logged to the server console, and when documents are automatically deleted from the log file. Note the following details:

- ▶ **logfilename:** The log database file name, usually log.nsf.
- ▶ **log_option:** Log options:
 - 1 = Log to the console
 - 2 = Force database fixup when opening the log file
 - 4 = Full document scan (as opposed to quick scan or open)

Tip: If the log option should be both 1 and 2, simply add the options and it becomes log option 3.

- ▶ **not_used:** Always set to zero; this parameter is not currently used.
- ▶ **days:** The number of days to retain log documents.
- ▶ **size:** The size of log text in event documents.

The default is Log=log.nsf, 1, 0, 7, 40000. The log file (log.nsf) is deleted in seven days and can contain up to 40,000 words. All log information is also sent to the console.

Log=log.nsf, 1, 0, 7, 40000

We recommend that you run with the default Log setting. If you decide to keep the log file for more days or it contains more words, functions that access or update the log file can decrease in performance.

Accessing the Notes Log database

The Notes Log Database (log.nsf) can be accessed or viewed by using one of these methods:

- ▶ From the Lotus Notes client workspace, click **File-> Database-> Open**. At the Open Database display, select the **server** and **Notes Log** (file name = log.nsf). Click **Open**.
- ▶ Start the Domino Administrator client and click the **Files** tab. To open the database, double-click **Notes Log** (filename=log.nsf).

Controlling what is logged in the database

The notes.ini settings affect the amount and type of information that appears in the Notes log database (log.nsf). The notes.ini file provides all of the initialization settings for a Domino server. Each time the server starts, it checks this file for information on which server tasks run and how to set up the server environment.

For monitoring performance, the following notes.ini settings are recommended:

► **LOG_AGENTMANAGER=1 (no default)**

Specifies whether the start of agent execution is recorded in the log file and shown on the server console:

- 0 Do not log agent execution events.
- 1 Log agent execution events.

► **LOG_MAILROUTING=10 (default is 20)**

Specifies the level of information recorded in the log file and controls other logging actions:

- 0 Defaults to 20.
- 10 Displays only the errors, warning, and major routing events, for example, startup and shutdown, number of messages transferred to x, and occurrences of database compacting. Successful deliveries and transfers are not recorded in the log file.
- 20 Same as for 10, except that successful deliveries and transfers are logged.
- 30 Displays thread information.
- 40 Displays transfer messages, message queues, and full document information for mail.box.

► **LOG_REPLICATION=2 (no default)**

Specifies whether the start and end of replication sessions are recorded in the log file and displayed on the console:

- 0 Do not log replication events.
- 1 Log server replication events.
- 2 Log replication activity at the database level.
- 3 Log replication activity at the level of database elements (views, document, etc.).
- 4 Log replication activity at the field level.
- 5 Log summary information.

► **LOG_SESSIONS=0 (no default)**

Specifies whether individual sessions are recorded in the log file and displayed on the console:

- 0 Do not log individual sessions.
- 1 Log individual sessions.

► **LOG_TASKS=1 (no default)**

Specifies whether the current status of server tasks is recorded in the log file and displayed on the console:

- 0 Do not send status information.
- 1 Send the status of server tasks to the log file and console.

► **LOG_VIEW_EVENTS=0 (no default)**

Specifies whether messages created when views are rebuilt are recorded in the log file:

- 0 Do not log messages when views are rebuilt.
- 1 Log messages when views are rebuilt.

► **LOG_UPDATE=0 (no default)**

Level of index logging:

- 0 Records when indexer starts and shuts down.
- 1 Records when the Indexer starts and shuts down + when Indexer updates views + full text indexes for specific databases.
- 2 As 1 + record names of views the Indexer is updating.

► **MAIL_LOG_TO_MISCEVENTS=0 (no default)**

Determines whether all mail event messages are displayed in the Miscellaneous Events view of the log file:

- 0** Does not display mail events in the Miscellaneous Events view.
- 1** Displays mail events in the Miscellaneous Events view.

► **NO_FORCE_ACTIVITY_LOGGING=0 (no default)**

Controls whether the Statlog task automatically enables activity logging on all databases:

- 0** Allows automatic activity logging on all databases.
- 1** Prevents automatic activity logging on all databases.

► **PHONELOG=0 (default 2)**

Specifies whether phone calls are recorded in the log file:

- 0** Does not record phone calls to the log file.
- 1** Records all calls, except those that fail because of a busy signal.
- 2** Records all phone calls.

► **SERVER_SHOW_PERFORMANCE=1 (no default)**

Specifies whether server performance events are displayed on the console:

- 0** Records server performance events in the log file.
- 1** Displays server performance events on the console.

► **SHOW_TASK_DETAIL=1 (no default)**

Specifies whether the name of the current executing transaction is added to the session message:

- 0** Do not include the name.
- 1** Include the name.

Figure 5-17 shows an example of the notes.ini settings.

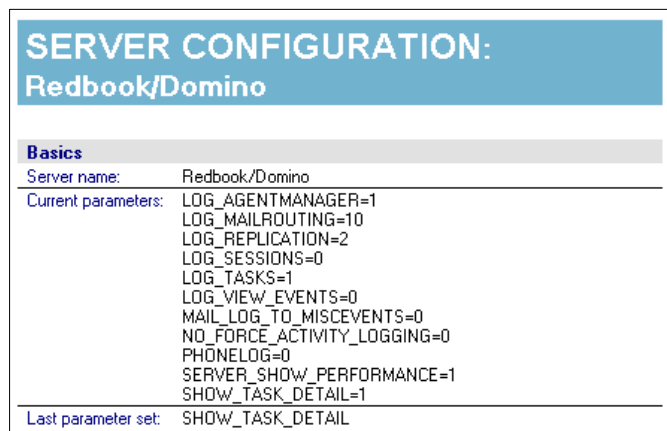


Figure 5-17 Server Configuration Document for controlling notes.ini settings

Note: The settings listed in Figure 5-17 pertain to logging statistics for performance. For problem determination work, you may need to change the settings.

Editing notes.ini settings

There are multiple ways that you can edit the notes.ini file. We outline the three most common ways here:

- ▶ Start the Domino administrator, and click the **Configuration** tab. Expand the **server** section. Edit or add the configuration document. Add or change to the notes.ini section of the configuration document.
- ▶ Select option **13** from the Work with Domino Servers (WRKDOMSVR) command to edit the notes.ini file for a particular server.
- ▶ Through Operations Navigator, the graphical user interface to the iSeries, you can work with the Domino servers on a given server. Right-click the server of your choice and choose **Properties**. Then, you can edit the notes.ini file.

Figure 5-18 shows an example of editing the notes.ini file using Operations Navigator.

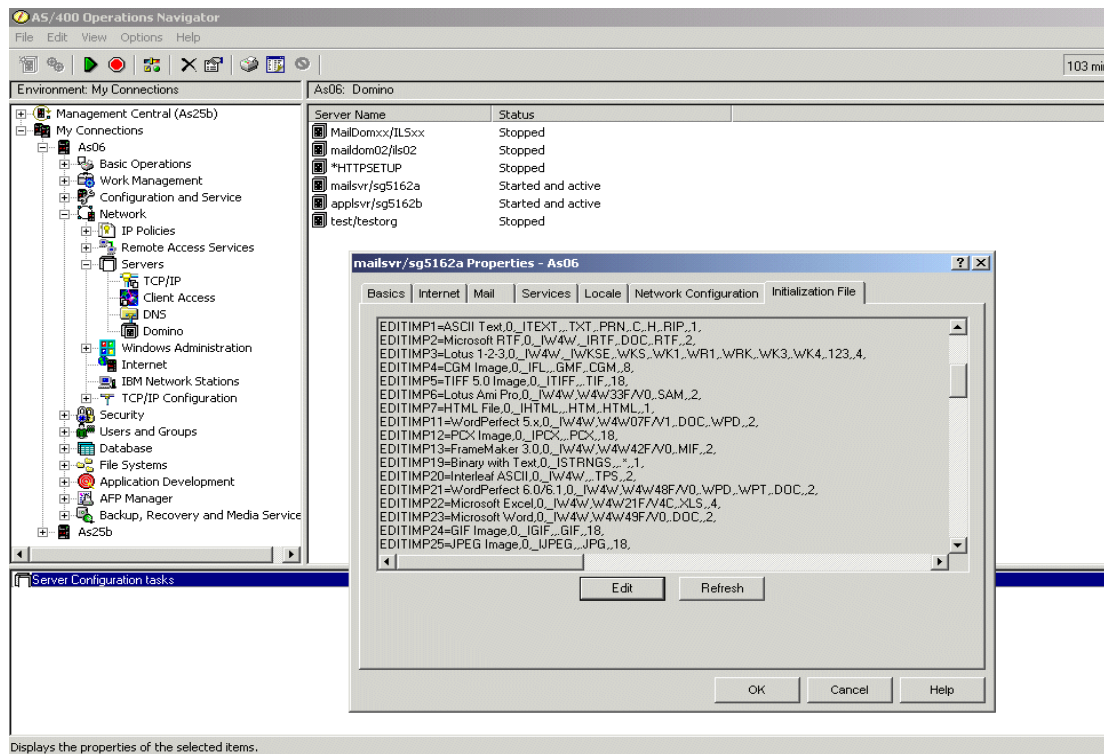


Figure 5-18 Editing notes.ini using Operations Navigator

5.2.2 The Statistics and Reporting database (statrep.nsf)

The Domino server continuously updates statistics. To view system statistics at any time, you use the **show stat** command at the server console, as explained in 5.1.2, “Understanding the Domino statistics” on page 132. To use statistics to monitor the Domino server over a period of time at a specified interval, you can use the Collect task, which collects statistics and places the information into the Statistics database (statrep.nsf).

The Domino server also provides monitor documents that you use to configure statistic thresholds. When the Collect task collects a statistic and places it into the Statistics database (statrep.nsf), it compares the statistic to the threshold configured in the monitor document. The first time the statistic reaches the specified threshold in the monitor document or in the Statistic Names document, an alarm is generated. An *Alarm report* is nothing more than a document informing you that a statistic has reached its threshold. To set up an alarm, you must specify a collection alarm interval in the Server Statistic Collection document. Alarm documents are automatically created in the Server - Analysis Tab - Statistics Reports - Alarms view in the Domino Administrator.

To notify you about important system information, the Domino server has a notification facility called the *Event task*. The Event task sends notification about the event to a destination configured in the Event Notification document.

You can create statistics and events monitoring documents for your particular server and network needs. To do this, you can create Statistic Monitors and Event Notification documents from the Statistics & Events view on the Configuration tab in the Domino Administrator, or create a new event notification in the database for Statistics & Events (events4.nsf). If you do not set up an Event Notification document, you will not be notified that an event has occurred.

Accessing the Statistics and Reporting database

The Statistics and Reporting database can be accessed or viewed by using one of the following methods:

- ▶ Start from the Lotus Notes client workspace, and click **File-> Database-> Open**. Then, select the **server** and **Statistics Reports** (file name = statrep.nsf). Click **Open**.
- ▶ Start the Domino Administrator client, and click the **Files** tab. To open the database, double-click **Statistics Reports** (file name=statrep.nsf).
- ▶ Start the Domino Administrator client and click the **Server** tab. Then, click **Analysis** and expand **Statistics** report.

Controlling what is logged in the database (statrep.nsf)

The following Domino server jobs (tasks) record events and statistics in the Statistics and Reporting Database:

- ▶ The Event server job
- ▶ The Collector server job

To configure which events to monitor and where to report them, create a Statistics Monitor Document and an Event Notification Document in the Statistics and Events Database (events4.nsf) for each type of event that you want to monitor.

Configuring statistics monitors and events

To add documents, perform the following steps:

1. Starting from the Lotus Notes workspace, click **File**.
2. Select **Database**.
3. Click **Open**.
4. On the Open Database display, select the **server** and **Statistics & Events Database** (file name = events4.nsf).
5. Click **Open**.
6. On the Statistics & Events display, select the **Statistic** view.
7. Under **Statistic**, click the **Monitors** view.
8. To add a statistic to monitor, click the **Add Statistic Monitor** button.

As shown in Figure 5-19, more than 700 users on the server can trigger a statistic server event of Warning (High).

Statistic Monitor - Server.Users	
If the statistic, on the server(s),	Number of users with open sessions on the server Any server
becomes Greater than	700 users
Generate a Statistic event of severity Warning (high)	
<input type="checkbox"/> Temporarily Disable Statistic Monitor	
Statistic Description:	

Figure 5-19 Statistic Monitor document example

To determine what to do with the statistic server event of Warning (High), you need an Event Notification document. To create the Event Notification document, perform the following tasks:

1. At the Statistics & Events display, select the **Event Monitors** view.
2. To add an event to the monitor, click the **Add Event Monitor** button.

Figure 5-20 shows an example of an Event Monitor document.

Event Notification	
If an event type, and severity, occurs on the server(s). Event notification method:	Statistic Warning (high) Any server Mail Enter mailing address: Redbook
<input type="checkbox"/> Temporarily Disable Event Monitor	

Figure 5-20 Event Notification document example

In Figure 5-20, a statistical event of Warning (High) mails a notification to the user “Redbook”.

The Collector server task

The Server Statistic Collection document in Statistics & Events database (events4.nsf) lets you set up a server that collects statistics from a list of specified servers. If you do not configure this document, the Collect task collects, by default, statistics only from the server that has the Collect task running on it.

To optimize performance, use the Collector server job to collect statistics of all servers because it impacts only one server in the environment. This frees the other servers in your environment for end-user access.

The content of the statistical reports is the same information you see by running the **show stat** command. The statistical report is basically a collection of **show stat** commands run over a period of time at a specified interval.

The Event server task

The Event server job runs by default. At the server console, you can stop the Event server task by entering the **tell event quit** command and start the Event server task by using the **load event** command.

To optimize performance, minimize the number of monitors used by the Event server task to meet your organization's requirement for monitoring server events. If your organization does not require server event monitoring, do not run the Event server task.

The Report task: The Report task no longer exists in R5. It has been replaced by the Collect task, which must be loaded manually or added to the ServerTasks line in the notes.ini. If you attempt to load the Report task, the error message appears: "Error attempting to load or run report.exe: Unable to locate program."

5.3 Combining performance data from Domino and iSeries

To combine performance data from the iSeries with the Domino server or servers, use the Statistic Collector to sample information from one or multiple Domino servers running on the iSeries server. For example, if you have two Domino servers on one iSeries server, you can use one of the servers to collect statistics from both the servers to the same statistics database. You can also monitor your whole Domino server environment performance combined with the iSeries performance data.

The statistic database and collector job is further described in 5.2.2, "The Statistics and Reporting database (statrep.nsf)" on page 149.

To get the statistics records, you must add PLATFORM_STATISTICS_ENABLED=1 to notes.ini. You must also create a Server Statistic Collection document where you define the collecting server and the server or servers to collect data from, as well as the database to receive reports and the collecting report interval. The Domino server Collect job must be running at the collecting server. You can read more about platform statistics in 5.1.4, "Platform-dependent statistics" on page 137.

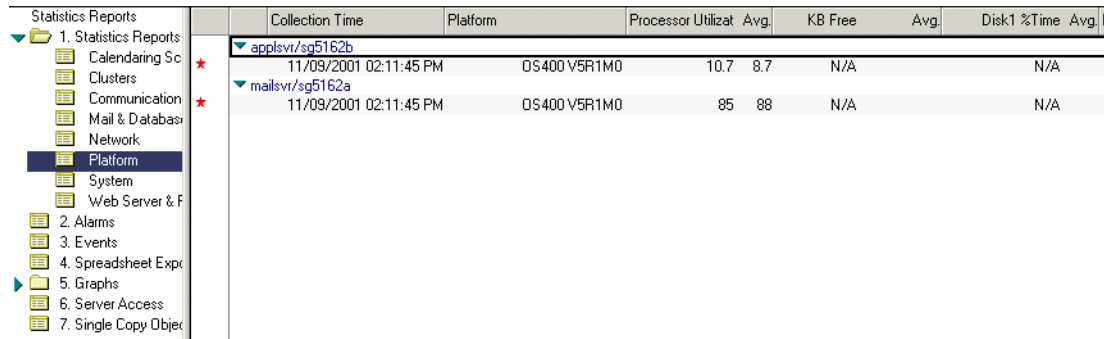
Figure 5-21 shows an example of the Server Statistic Collection document. It has been configured to collect statistics from two Domino servers at two different domains, with the collecting server applsvr/sg5162b. Both servers are on the same iSeries.

Statistics & Events		New Server Statistic Collection			
<ul style="list-style-type: none"> Setup Wizards Event Notification Monitors <ul style="list-style-type: none"> ACL Change File Replication Statistic Probes <ul style="list-style-type: none"> Domino Server Mail TCP Server Server Statistic Collection Names & Messages (Advanced) 		Server	Collected By	Report Interval	Alarm Interval
		applsvr/sg5162b	applsvr/sg5162b	5 minutes	15 minutes
		mailsvr/sg5162a			
			applsvr/sg5162b	5 minutes	15 minutes

Figure 5-21 Server Statistic Collection document

Modifying the Statistics Reports database for iSeries and Domino data

In the Statistics and Reports Database (statrep.nsf), there are standard views that come with the Domino application. If you select to look at the Platform view, the only OS/400 statistics available are Platform.System.TotalUtil and Platform.System.TotalUtil.avg, as shown in Figure 5-22.



Collection Time	Platform	Processor Utilizat. Avg.	KB Free Avg.	Disk1 %Time Avg.
11/09/2001 02:11:45 PM	OS400 V5R1M0	10.7 8.7	N/A	N/A
11/09/2001 02:11:45 PM	OS400 V5R1M0	85 88	N/A	N/A

Figure 5-22 Statistics and Reports Database: Standard views

To view more OS/400 platform statistics in the statistics database, you can modify the Platform view or a better way is to create one or more additional views.

To prevent your modification from being lost by a replace design or upgrade, we recommend that you:

1. Create a new copy of the Statistics and Reports Database template (statrep5.ntf).
2. Name the new copy *statrep400.ntf*.
3. Do the modifications to your template.
4. Create your own statistics database based on statrep400.ntf.
5. Configure the Statistics Collection document to use your statrep400 database as the receiver.

Here, we show an example of how you can create an additional view to your new database template, statrep400.ntf, that shows OS/400 platform statistics and Domino statistics together. The intent is to give you an idea of how you can modify the views to gather different sets of statistics and to provide you with performance data that you need in your environment.

1. Start the Domino Designer and open the database template statrep400.ntf.
2. Create a copy of the Statistics Reports\Platform view.
3. Rename the copy to iSeries Platform Statistics\iSeries and Domino Statistics.
4. Open the new view for design changes, remove the fields you don't need in this view. Add OS/400 statistics fields (Platform.x.x.x.) and Domino fields that you want to have in the same view.
5. Save your changes.

Section 5.1.4, "Platform-dependent statistics" on page 137, provides information about the OS/400 statistics fields (Platform.x.x.x.) that are available and can be added to the view.

For detailed information about how to create/modify database views, see the *Domino Designer Help Guide*.

Figure 5-23 shows you the statrep400.ntf template opened in Domino Designer.

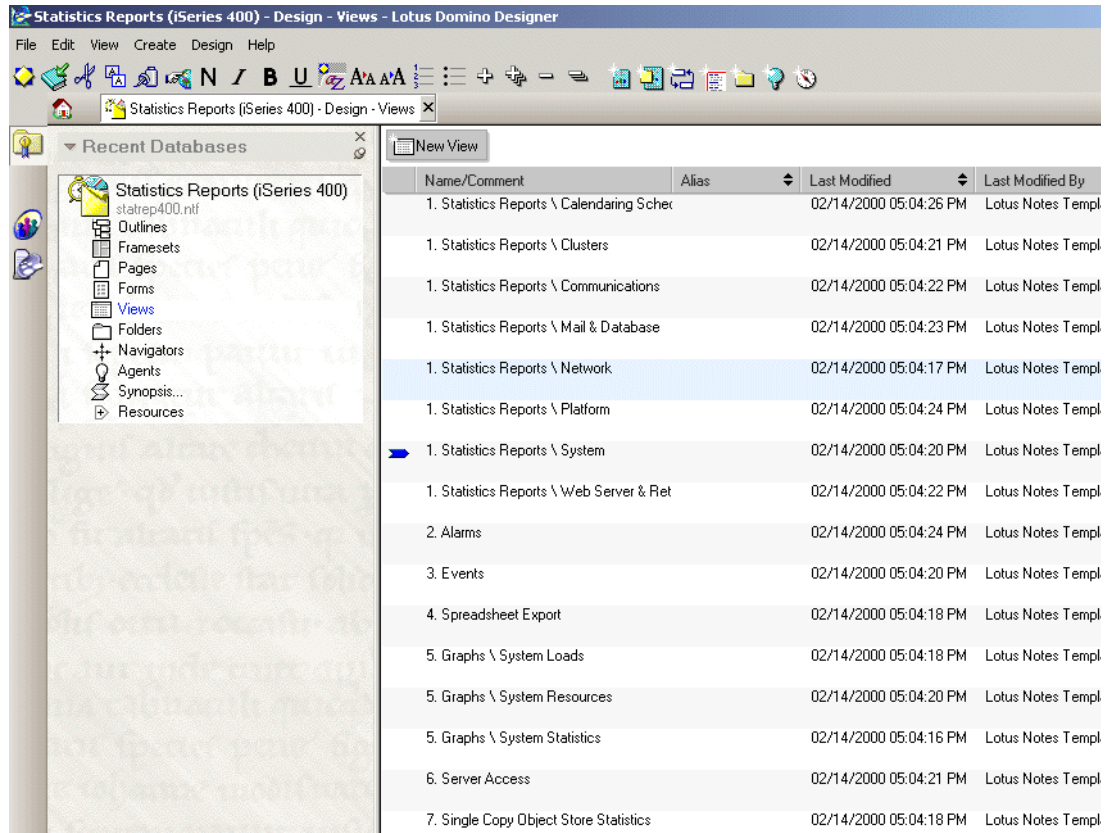


Figure 5-23 Statrep400.ntf template: Opened in Domino Designer

Figure 5-24 shows you the statrep400.ntf template with the newly copied and the renamed view *iSeries Platform Statistics\iSeries and Domino statistics*.

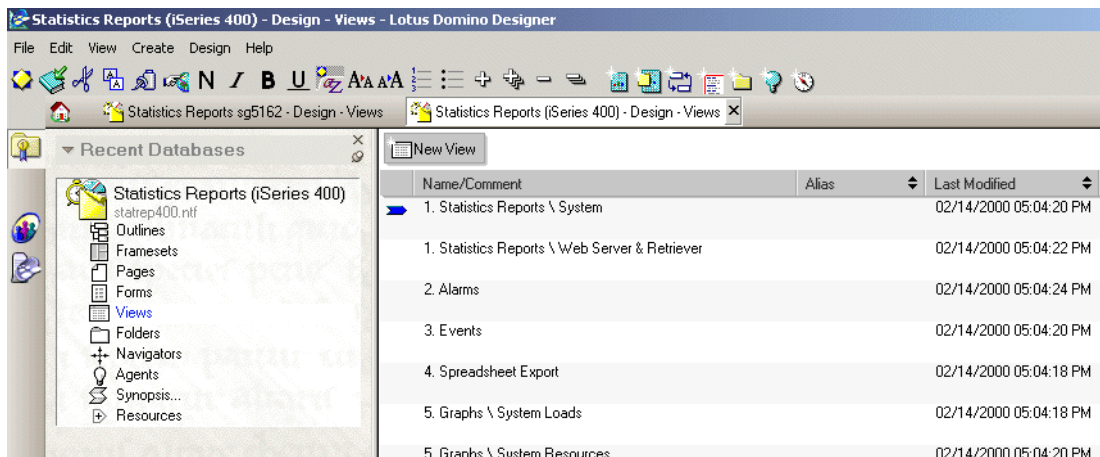


Figure 5-24 Statrep400.ntf with new *iSeries Platform Statistics\iSeries and Domino statistics* view

Figure 5-25 shows you the Statistics and Report template, statrep400.ntf, with iSeries Platform Statistics and Domino statistic fields. The Transactions per minute and Active users are Domino statistics, and the rest are OS/400 statistics fields.

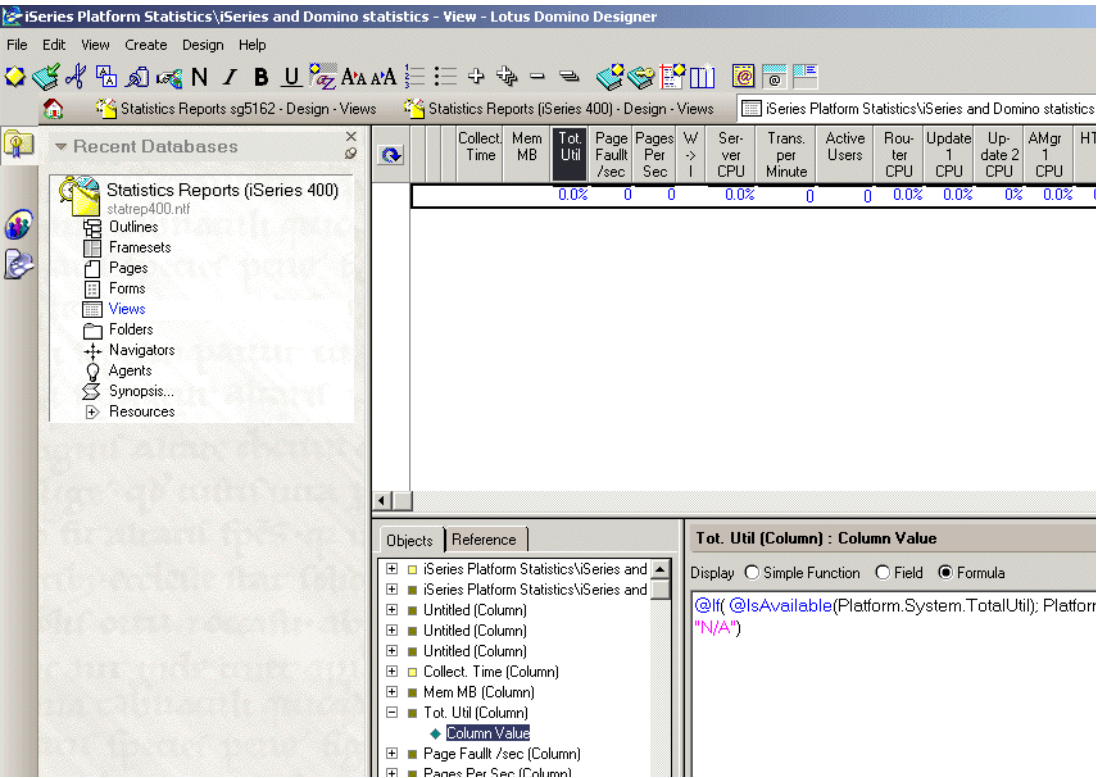


Figure 5-25 Statrep400.ntf with iSeries Platform Statistics and Domino statistic fields

In Figure 5-26, you see the Statistics and Report database, statrep400.nsf, with iSeries Platform Statistics and Domino statistic data. The transactions per minute and active users are Domino statistics, and the rest are OS/400 statistics data.

The screenshot shows the Lotus Notes interface for the 'Statistics Reports' database (statrep400.nsf). The interface includes a menu bar (File, Edit, View, Create, Actions, Help) and a toolbar. On the left, there is a 'Statistics Reports' pane showing the database structure, including '1. Statistics Reports', '99. GROUP Tools', 'Calendaring Sched...', 'Clusters', 'Communications', 'Mail & Database', 'Network', 'Platform', 'System', 'Web Server & Retri...', '2. Alarms', '3. Events', '4. Spreadsheet Export', '5. Graphs', '6. Server Access', '7. Single Copy Object St', 'Wilfried's AS/400 Statisti', and 'Current AS/400 Stats'. The main area displays a table with columns for various statistics: Coll. Time, Mem MB, Tot. Util, Page Fault /sec, Pages Per Sec, Disk Util, Free Disk (GB), W. I, Ser-ver CPU, Tran-s per Minute, Act. User, Router CPU, Upd. 1 CPU, Upd. 2 CPU, AMgr 1 CPU, and HTTP 1. The table shows data for various statistics, including '5.0.8', 'V4R5M0', 'V5R1M0', 'FMS170MBI', and 'vWe 12.1'.

	Coll. Time	Mem MB	Tot. Util	Page Fault /sec	Pages Per Sec	Disk Util	Free Disk (GB)	W. I	Ser-ver CPU	Tran-s per Minute	Act. User	Router CPU	Upd. 1 CPU	Upd. 2 CPU	AMgr 1 CPU	HTTP 1
5.0.8		510	22.6%	2	8	1.2	22.3	6.1%	16	2	0.1%	2.1%	0%	0.1%	0.0%	
V4R5M0		310	54.2%	0	0	2.5	29.5	16.4%	5	1	0.4%	0.3%	0%	0.4%	0.0%	
V5R1M0		604	7.8%	2	11	0.5	18.9	1.3%	21	2	0.0%	2.9%	0%	0.0%	0.0%	
FMS170MBI		604	7.8%	2	11	0.5	18.9	1.3%	21	2	0.0%	2.9%	0%	0.0%	0.0%	
vWe 12.1		538	9.7%	1	8	0.3	18.6	1.4%	4	3	0.0%	5.1%	0%	0.0%	0.0%	
08:55		554	4.7%	0	1	0.1	18.7	0	1.6%	8	3	0.0%	0.0%	0%	0.0%	0.0%
08:25		566	3.3%	1	3	0.4	18.7	0	1.6%	4	2	0.0%	0.0%	0%	0.0%	0.0%
07:55		485	5.0%	0	1	0.8	18.8	0	1.6%	2	1	0.0%	0.0%	0%	0.0%	0.0%
07:25		528	93.9%	17	114	2.0	18.8	0	1.6%	4	1	0.0%	81.3%	0%	0.0%	0.0%
06:55		525	5.0%	0	1	0.0	18.8	0	1.6%	4	1	0.0%	0.0%	0%	0.0%	0.0%
06:25		524	3.3%	0	2	0.2	18.8	0	1.6%	4	1	0.0%	0.0%	0%	0.0%	0.0%
04:55		557	4.6%	0	0	0.1	18.6	0	1.6%	4	2	0.0%	0.0%	0%	0.0%	0.0%
04:25		507	4.5%	0	0	0.1	18.6	0	1.6%	4	2	0.0%	0.0%	0%	0.0%	0.0%
03:55		491	4.5%	0	0	0.2	18.6	0	1.6%	4	4	0.0%	0.0%	0%	0.0%	0.0%
03:25		506	4.5%	0	0	0.1	18.6	0	1.6%	4	5	0.0%	0.0%	0%	0.0%	0.0%
02:55		566	3.5%	0	0	0.2	18.6	0	0.0%	4	5	0.0%	0.0%	0%	0.0%	0.0%
02:25		586	3.6%	0	1	0.2	18.6	0	1.6%	4	5	0.0%	0.0%	0%	0.0%	0.0%

Figure 5-26 Statistics and Report database with combined iSeries and Domino statistics

5.4 Monitoring the Notes Remote Procedure Calls (NRPC)

How do you know what actually is happening at the network level with *Notes Remote Procedure Calls* to and from a workstation. How can you monitor the actual traffic and know if your proposed application will actually scale, or indeed whether your network can take the strain? This section answers these questions.

5.4.1 Using a network sniffer

One option to help you trace frames and packets is to use a network sniffer. A remarkably good one is available as shareware from: <http://www.ethereal.com>

Ethereal

Ethereal is a GUI network protocol analyzer. It lets you interactively browse packet data from a live network or from a previously saved capture file. Ethereal knows how to read libpcap capture files, including those of tcpdump.

In addition, Ethereal can read capture files from:

- ▶ Snoop (including Shomiti) and atmsnoop
- ▶ Lanalyzer
- ▶ Sniffer (compressed or uncompressed)
- ▶ Microsoft Network Monitor
- ▶ AIX's iptrace
- ▶ NetXray
- ▶ Sniffer Pro
- ▶ Etherpeek
- ▶ RADCOM's WAN/LAN analyzer
- ▶ Lucent/Ascend router debug output
- ▶ HP-UX's nettl
- ▶ The dump output from Toshiba's ISDN routers
- ▶ The output from i4btrace from the ISDN4BSD project
- ▶ The output in IPLog format from the Cisco Secure Intrusion Detection System
- ▶ Pppd logs (pppdump format)

There is no need to tell Ethereal what type of file you are reading; it determines the file type by itself. Ethereal is also capable of reading any of these file formats if they are compressed using gzip. Ethereal recognizes this directly from the file; the ".gz" extension is not required for this purpose.

Three-view window (whole, line, and hex)

Like other protocol analyzers, Ethereal's main window shows three views of a packet. It shows a summary line that briefly describes what the packet is. A protocol tree is shown that allows you to drill down to the exact protocol or field in which you are interested. Finally, a hex dump shows you exactly what the packet looks like when it goes over the wire.

Multiple and DIY display filters

In addition, Ethereal has some features that make it unique. It can assemble all the packets in a TCP conversation and show you the ASCII (or EBCDIC, or hex) data in that conversation. Display filters in Ethereal are very powerful; more fields are filterable in Ethereal than in other protocol analyzers, and the syntax you can use to create your filters is richer. As Ethereal progresses, expect more and more protocol fields to be allowed in display filters.

Packet capturing and compressed file support

Packet capturing is performed with the pcap library. The capture filter syntax follows the rules of the pcap library. This syntax is different from the display filter syntax. Compressed file support uses (and therefore requires) the zlib library. If the zlib library is not present, Ethereal will compile, but will be unable to read compressed files.

5.4.2 Lotus Client NRPC Monitoring Tool Client_Clock

Notes also has its own built-in monitoring capability. This is an unsupported, but reasonably well documented utility, that is well-known as a useful debug parameter. This parameter analyzes traffic to transaction level and shows the duration in seconds.

For more information, see Technote 163971 and this site regarding Client_Clock:

http://domino-8.prominic.com/A55711/ref/notesini.nsf/5a0c8fb630f30a6bc12569a1005f807f7a4872639fb90313c12567d70052ce32!OpenDocument&Highlight=0,CLIENT_CLOCK

Important: Before you enable any debug parameters of this type, it is imperative that you test them out on a Test system and test network. There may be issues surrounding their use or special precautions that you must consider. Debug parameters often require a large amount of disk space, dependent upon traffic volumes. Lotus could also change the code without notice completely altering the affect and implications of use.

Overview of NRPC Monitor

The Notes Client uses NRPCs to communicate with the server. Ping-ponging RPC calls (repeating back and forth) are expensive because they generate network traffic as well as unnecessary server load. The goal is to reduce as much ping-ponging RPC traffic within the design of your application as possible, shorten the data payload lengths, and reduce the frequency that you send packets in the first place. In summary, the goal is to minimize volume and frequency.

Setting up monitoring

To monitor RPC traffic from a Notes workstation, add three new lines to the notes.ini file as shown in Table 5-1. The table shows the three client notes.ini variables you need to set to enable and capture NRPCs.

Table 5-1 Unsupported Client_Clock parameters and syntax

Parameter	Description
Client_Clock = 1	Activates N-RPC tracing
Debug_Console = 1	DOS Window monitoring real-time traffic
Debug_Outfile=<path\filename>	Pipes the trace output to a text file

On startup, you see two boxes, the console and the re-directed log, with some additional information for example:

(54-997) NIF_OPEN_NOTE: 2119 ms. [48+9370=9418]

Table 5-2 shows a quick breakout to help you interpret the results of NRPC tracing.

Table 5-2 RFC command stream breakout and some additional comments

De-Code:	Meaning
(54-997)	Sequence Number
NIF_OPEN_NOTE=	Interpretation of the call
2119 ms.	Time in milliseconds to complete the call
48	Number of bytes sent
9370	Number of bytes returned
9418	Total of bytes for this call
[48+9370=9418]	Data sent+data received=total data in bytes

There are a lot of different things that happen between the client and the server. However, if you follow the commands, and you know the code you wrote, then as with any other application debugger, you should be able to make sense of it.

Here are some general hints and tips for monitoring NRPC or any Network traffic:

- ▶ Simplify what you are trying to trace. Start easy and work your way to the more complex items as you become familiar with the process.
- ▶ Disable workstation replication to stop any unwanted NRPCs from being merged.
- ▶ Pause between each key stroke or action so that you can separate/debug commands.

Remember that the tighter the code is, the less traffic there is and the better the performance is.

We now analyze an example involving LotusScript code in the QueryClose event of a form. This code opens a view in the current database, retrieves a document by key in the current database, and finally creates a new document in another database. Table 5-3 shows the LotusScript statements issued along with the resulting series of NRPCs.

Table 5-3 LotusScript showing as RPC commands using Client_Clock

LotusScript statement	RPCs
Set MyView= db.GetView("LookupView")	(332-1952) OPEN_COLLECTION: 514 ms. [32+612=644]
	(333-1953) READ_ENTRIES: 2143 ms. [46+7254=7300]
	(334-1955) CLOSE_COLLECTION: 0 ms. [12+0=12]
	(335-1955) OPEN_NOTE: 1538 ms. [28+2502=2530]
	(336-1956) OPEN_NOTE: 1273 ms. [28+2502=2530]
Set MyDoc= MyView.GetDocumentByKey("key", True)	(337-1958) FIND_BY_KEY: 356 ms. [32+34=66]
	(338-1958) READ_ENTRIES: 411 ms. [46+94=140]

LotusScript statement	RPCs
	(339-1959) OPEN_NOTE: 425 ms. [28+342=370]
Call newDoc.Save(True,True)	(340-1959) OPEN_DB: 459 ms. [126+210=336]
	(341-1959) DB_INFO_GET: 640 ms. [14+140=154]
	(342-1960) UPDATE_NOTE: 557 ms. [566+56=622]
	(343-1961) CLOSE_DB: 0 ms. [14+0=14]

For more information, see *Performance Considerations for Domino Applications*, SG24-5602. You should also see 5.1.5, “Show Domino Transactions” on page 143.



Tuning the iSeries server for Lotus Domino

With Domino and the iSeries working together, there are many different variables that can affect overall performance. To help you understand the full picture, we need to address both Domino and iSeries performance.

This chapter is designed to address the OS/400 perspective of the equation so you can have a full understanding of how to analyze and influence the performance of Domino applications and functions running on the iSeries server. In Chapter 7, “Tuning Lotus Domino for better performance on iSeries” on page 183, you can learn how the Domino administrator can effect the throughput of one or more Domino servers running on an iSeries server.

This chapter includes these topics:

- ▶ Overview of resource tuning the iSeries with respect to special behaviors of Lotus Domino
- ▶ OS/400 automatic performance adjustment
- ▶ When to use separate memory pools
- ▶ Setting the run time priority of Domino tasks
- ▶ Using online backup with BRMS
- ▶ Verifying the configuration of your network

6.1 Tuning the iSeries server

Working on and fixing a performance problem is similar to solving a three-dimensional puzzle. All parts of the puzzle initially appear to be unconnected. Changing one part (for example, increasing a priority) seems to cause unexpected changes and problems with what was thought of as a totally unconnected part of the whole puzzle! But, after solving several such puzzles, you begin to grasp a pattern. Solving a Domino performance problem is a challenging task. It requires an open mind and a “holistic” approach (exclude nothing from your initial enquiries). It also requires the ability to admit your earlier hypothesis may have been wrong and you may need to start with a new line of thinking. The most common cause of failure in diagnosis is to choose the wrong path and never come back for another look around.

Before you attempt to fix or tune Domino performance issues, you must have a solid base to work from, that is the platform, which in this case is the iSeries server. The base platform itself must be tuned properly (as far as you can) before you can even start to determine what is causing the Domino performance problem.

Domino is simply an application suite that has “work” and “memory management” functions of its own and is designed (on the iSeries) to run in a dedicated subsystem of its own. Tuning Lotus Domino for iSeries and tuning the iSeries server, therefore, have to go together. First, tune the iSeries server to provide Domino with (what you think are) the resources it needs. Then tune Domino. Then perhaps come back and re-tune the iSeries a little more. And then possibly go back to re-tune Domino again and so on until everything is balanced and running in harmony. This chapter explains how to provide those iSeries server resources to the Domino application.

Any actual tuning (changes to) the iSeries server should only be attempted after a very careful analysis of the overall performance characteristics of the system is completed. This is described in Chapter 4, “Basic concepts of performance analysis” on page 53.

Tuning the iSeries server involves these steps:

1. Measure the performance.
2. Change one parameter or setting at a time.
3. Measure the performance again, compare the results, and go back to step 2, if you are not satisfied yet.

Tuning the system means that you try to allocate the available system resources to users or batch jobs that are in most need of those resources. Remember, there is an absolute and finite limit on the number of resources in the system, and all the jobs in the system compete for available resources. Once all resources have been used up, all “tuning” really achieves is to take resources away from one job and give them to another. You cannot increase available resources, obviously, but only re-allocate them.

Hopefully you will find one job or subsystem over-allocated with spare resources, and you can safely re-allocate those spare resources to a job that is starved of resources and everyone is happy. But eventually, once all jobs have the correct resources that they need, you will be at the point where one job is stealing resources from another. That is the time when you must upgrade the hardware to gain additional resources. In truth, you should have seen this coming from your monitoring of the system *before* the workload grew large enough to eat up all the available system resources.

System tuning also involves making every attempt to minimize or eliminate the unnecessary system work, for example:

- ▶ Monitor the number and arrival rate of messages in the job logs and history logs.
- ▶ Monitor all error logs in the service tools (SST).
- ▶ Avoid multiple sign on and sign off functions.
- ▶ Clear all output queues, old history files, and all old journal entries on a regular basis.

Important: When making changes, apply only one change at a time and measure the effect of that change on system performance.

Perhaps the easiest way to affect the system performance is to evenly distribute the system's workload, for example:

- ▶ Move heavy batch jobs to run during the night or other off shift time.
- ▶ Perform system save activities during a lower system usage time.
- ▶ Assign the programmers a subsystem and storage pool of their own, reserved only for their work.
- ▶ Keep the workload on the system as constant as possible.

Once you have tried re-distributing the workload and tuning as much as possible, and it still does not meet your performance goals, it's time to buy more hardware!

Upgrading your system's hardware may seem like an expensive solution. But when you consider the costs of hiring and maintaining a staff with the required skills to streamline your applications (and with the skills to perform a Performance Audit and produce a concise but accurate report), the cost of upgrading the system hardware may actually be lower in comparison. If you follow the advice and guidance in this redbook, you will know and understand the available options and how to measure and justify your decisions.

6.2 Automatic performance adjustment (QPFRADJ)

The automatic performance adjustment in OS/400, activated and controlled through the system value QPFRADJ, can be a great boon and a great tool, but in very few situations, it may have a negative impact. In general terms, it is a fine tool to leave switched on; it balances workloads against resources and constantly strives to optimize the performance of all jobs and pools on the iSeries server. It is like having a fully skilled OS/400 expert constantly fiddling and re-balancing your system. Having said that, its very innate ability can also be its own downfall where Domino is concerned.

The main issue involved in having the automatic performance adjuster turned on is when the workload on the system changes over the course of a day. Here is a typical customer scenario that describes the situation:

1. The workload during the prime shift involves Domino applications that provide various services for the end users. The Domino server may route mail, perform workflow functions designed into the application, access DB2 UDB data, and serve the company's Web site.

While all of this is happening, the automatic performance adjuster looks at all of the work happening on the system and provides Domino with the adequate resources. This involves increasing the activity level of the memory pool where Domino is executing and allocating more memory to the jobs that support the Domino function.

2. In the evening, the workload on the system changes. Most of the Domino activity ceases, and other night time tasks fire up, such as backups, batch jobs, and other clean-up tasks. The automatic performance adjuster now moves memory and activity levels to these tasks so they have the appropriate resources.

3. The situation arises in the morning when all of the workers come into work and start using the Domino applications and services again. At this point, the automatic performance adjustor needs to start moving all of the memory from the night tasks back to the Domino server. This can mean poor performance until the automatic performance adjustment has caught up in rebalancing resources.

Performance adjustment is relatively slow to react to such a sudden change (logically that is correct; it shouldn't suddenly rush to allocate massive resources into the first spike of activity it detects). Because of this, it only re-establishes the resources over a period of some minutes. Meanwhile every Notes Client is attacking the Domino server with a connection request. In R5, if Domino doesn't give a quick response, the client generates another request to the server, and then another, up to 5, in an attempt to establish a server session.

As hundreds of users fire off these multiple connection attempts ("retries"), Domino will inevitably belatedly establish multiple connections for each user. Each Domino connection uses considerable server resources just to actually establish the session. In turn, this further ramps up the server load, slowing overall processing yet more, and therefore, causing other users to issue multiple connects in an ever increasing downward spiral.

Domino itself also has a self-protection mechanism to limit maximum user connections with the *Server_MaxUsers* parameter in notes.ini. However Domino can easily handle its standard workload of hundreds or thousands of users, with resources to spare. Every Notes client is now pretending to be up to five users, before long *Server_MaxUsers* is reached, and these connections will stay engaged until *Server_connection_timeout* (four hours by default¹) is reached!

So now your early morning users are being refused access, and your phone starts to ring (and continues to ring *Server_MaxUsers* times for *Server_Connection_Timeout* minutes to be exact!).

A simple solution to beat the "early morning rush" is to use the Work with Shared Pools (WRKSHRPOOL) command, press F11 (Display tuning data), and set a minimum size for the storage pool where your Domino servers are executing. Typically this is the *BASE pool. See 6.4, "Choosing which memory pool to use" on page 166, for information on using other pools.

Both a minimum and maximum percentage of total main storage can be specified for each memory pool. These settings are used by the system if the automatic performance adjustment system value QPFRADJ is set to 2 or 3. This will guarantee a minimum amount of memory will remain allocated to Domino, therefore, avoiding the early morning ramp up problem.

We recommend that you set QPFRADJ to 3 (Automatic adjustment). Option 2 (Adjustment at IPL and automatic adjustment) may also work to initially set the environment on a new system, but the values will be reset after each IPL and need to be re-adjusted after starting the servers.

6.2.1 The interactive iSeries tuning chart

In many cases, we recommend that you leave performance tuning to the system with the system value QPFRADJ set to 3. However, in special cases, it may be beneficial to do some or all of the tuning manually. If you decide to do so, follow our road map shown in Figure 6-1.

¹ The default to that is 4 hours, which later we recommend that you lower to 30 minutes.

To follow the road map, follow steps 1 through 3 to ensure the machine pool has sufficient resources. If the machine pool is starved for resources, performance of the whole system may be impacted. After determining that the machine pool has adequate resources, repeat steps 4 through 7 in Figure 6-1 for all of the other pools in your iSeries server. Do this during periods of high system activity because it has no value to tune the system when there is only a relatively light workload on the system.

Make sure that system value QPFRADJ is set to zero (to not adjust), before you follow the tuning road map. If you leave QPFRADJ set to value 2 or 3, it may undo the tuning you are attempting to perform.

1. Enter the WRKSYSSTS command. Press PF21 to set the assistance level to Intermediate.
2. Wait 2 - 3 minutes and press PF5 to refresh.
3. Do the *MACHINE NDB faults meet the guidelines?
 Yes..... Press PF10 and go to step 4
 No.....Adjust QMCHPOOL
 -500K if fault rate = 0
 +500K if fault rate >3.0
 Press PF10 to reset and go to step 2
4. Is the DB fault + NDB fault > 20 in any pool?
 Yes ... Increase pool size by 500KB, press PF10 and repeat step 4 (repeat until all pools are less than 20)
 No ... Go to step 5
5. Wait 2 - 5 minutes, and press PF5. Press PF21 to set the Assistance level to Advanced.
 Is the Wait-to-Ineligible state = 0 ?
 Yes ... Reduce Activity level by 2, press PF10 to reset and repeat step 5
 No Go to step 6
6. Is the Active to Wait state 10x the activity level?
 NoSystem not heavily used or complex application mix; go to step 4
 Yes ... Go to step 7
7. Is the sum of all fault rates for all pools within the guidelines?
 No Go to step 4
 Yes ... Go to step 8
8. Activity levels and pool sizes probably OK. Continue monitoring the WRKSYSSTS display regularly.

Figure 6-1 Interactive tuning roadmap

6.2.2 Summary

Automatic performance adjustment is extremely useful. iSeries servers work and stay working at their optimum with it switched on (option 3) and left on. Only on very large sites where a dedicated team of operators and system programmers permanently monitor and perform the same functions “by hand” can it be safely left off. Be sure you have looked very carefully at all the implications of not using this important iSeries feature if you choose not to use it.

Note: WRKSYSVAL QPFRADJ is not the only performance adjuster tool available for iSeries. Several IBM Business Partners have built similar and often more comprehensive toolkits that provide automatic performance monitoring and adjustment to provide a total and fully tailorable performance adjusting and tuning solution.

6.3 Determining Domino workload

Domino by itself is not an application. Domino is an Application Architecture Framework onto which you can hang an almost unlimited type and style of application program/job/task written in LotusScript, Java, and the Lotus command language (known as @Functions or macro language).

On the iSeries server, Domino is implemented in the subsystem architecture. Each Domino server then has its own set of tasks or jobs that comprise the function configured for a given Domino server. Each of the Domino tasks are presented as an iSeries job. For example, the SERVER task is implemented as a job called SERVER² in the Domino subsystem. This one job will spawn multiple threads to handle simultaneous requests from multiple clients.

Likewise, the HTTP Domino task is implemented as the HTTP job in the Domino server subsystem. The HTTP task will have multiple threads to handle the requests from various browser users.

You can find more detailed information about each Domino tasks in Chapter 8, “Understanding the Domino server jobs” on page 229. With this understanding, it becomes relatively easy to figure out what your Domino workload is. The Domino work will execute in Domino subsystems and each of these subsystems will have Domino tasks or jobs that spawn multiple threads to handle the incoming requests.

In the remainder of this chapter, you will find information about working with these various tasks or jobs. You’ll also find recommended settings for the various Domino parameters that you can tweak in your efforts to create the highest performing Domino environment on your iSeries server.

6.4 Choosing which memory pool to use

A memory pool is a logical division of main memory or storage that is reserved for processing a job or group of jobs. In OS/400, all main storage can be divided into logical allocations called *memory pools*. By default, the system manages memory pools. The system manages the transfer of data and programs into memory pools if necessary.

Note: Prior to OS/400 V5R1, the pools were called *storage pools*. Now Operations Navigator, the graphical user interface for OS/400 administrators and operators, uses the term memory pool, while most of the CL commands still refer to “storage pool”. To be consistent with the latest version of the user interface, we use the term “memory pool” in this book.

You can control how much work can be done in a subsystem by controlling the number and size of the memory pools. The greater the size of the memory pools in a subsystem, the more work can be done in the subsystem.

² There is not always a one-to-one match between the name of server tasks and the job name. The INDEXER task appears as an UPDATE job, and all job names starting with the letter Q (for example QNNINSTS) do not appear in the Domino tasks list.

By default, all Domino servers will run in the *BASE memory pool. This is quite adequate in most scenarios, but there are times when you may want to place a Domino server in its own storage pool to allow you to specify specific priorities, amounts of memory, and so on.

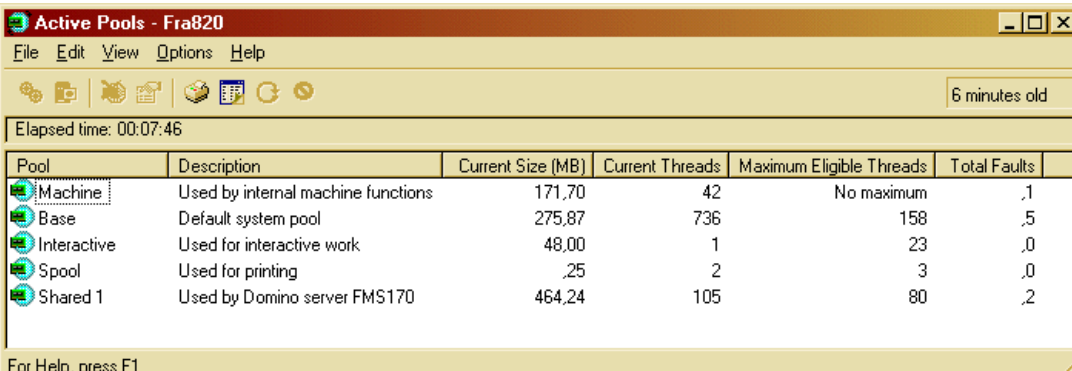
The scenario where this becomes most important is when Domino is running on the same server that is also hosting interactive, 5250-based applications written in RPG or other languages. Both the Domino tasks and the 5250 jobs will run at run priority 20. They will compete for the same CPU cycles. Since Domino tends to require much more processing power than traditional OS/400 applications, a heavy Domino application may severely impact the response times of other applications.

If Domino and traditional interactive applications were left to run at the same priority level and in the same memory pool, neither set of applications would run well. In this scenario, it is much better to separate the memory pools for the Domino server and interactive jobs.

The reason for doing so is this, as described above, is that the interactive jobs typically stay active for a very short time. If Domino tasks stay active for much longer time periods, the pages needed by the interactive jobs will be moved out of memory over time. Once the jobs become active again, those pages need to be moved back into memory. This costs valuable time and increases the response times.

The purpose of separate memory pools is to guarantee a certain amount of memory to a group of jobs, which will not be paged out, because other jobs (not part of the same group) need memory. Typically this “group of jobs” belongs to the same application or run in the same subsystem, but that is not necessarily so. For example, you may use the predefined shared³ pool *INTERACT for all your interactive 5250 applications.

Figure 6-2 shows an example of all active memory pools as they appear under Operations Navigator.



Pool	Description	Current Size (MB)	Current Threads	Maximum Eligible Threads	Total Faults
Machine	Used by internal machine functions	171.70	42	No maximum	.1
Base	Default system pool	275.87	736	158	.5
Interactive	Used for interactive work	48.00	1	23	.0
Spool	Used for printing	.25	2	3	.0
Shared 1	Used by Domino server FMS170	464.24	105	80	.2

Figure 6-2 Active memory pools

Periodically checking the amount of memory your memory pools use is important. By monitoring these levels, you can tune your pools to run at maximum efficiency, which in turn, keeps the work cycle running smoothly. In Operations Navigator, you can easily monitor the amount of memory your pools are using.

To check the memory usage, follow these steps:

1. In Operations Navigator, expand **My AS/400 Connections**.
2. Expand the connection for your **iSeries server**.
3. Expand **Work Management**.

³ “Shared” means this pool may be shared by jobs running in multiple subsystems, but not by all jobs in the system.

4. Expand **Memory Pools**, and then click **Active Pools** or **Shared Pools**.
5. Right-click the memory pool you want to work (for example, Interactive) and select **Properties**.
6. Click the **Configuration** tab. The Current field, under Size, shows the amount of memory the pool currently has.

The following sections describe the different types of memory pools and how to assign one to Domino servers.

The machine pool (*MACHINE)

This is the pool in which highly shared OS/400 jobs and microcode tasks (License Internal Code) run. The machine pool provides storage for jobs the system must run that do not require your attention. The size for this memory pool is specified in the system value QMCHPOOL. No user jobs run in this pool.

While Operations Navigator refers to the pools by name (see Figure 6-2), the machine pool appears as system pool identifier 1 on the Work with System Status (WRKSYSSTS) display. Normally, this pool should have the rate of non-DB faults less than one per second.

The base pool (*BASE)

This is the memory pool where all Domino jobs run by default. It contains all memory that is not allocated by other (private or shared) memory pools.

The base pool contains storage that can be shared by many subsystems. The system value QBASPOOL specifies the minimum size of the base pool. Its activity level is specified in the system value QBASACTLVL.

The base pool is used for batch work and miscellaneous system functions. While the Operations Navigator refers to the pools by name (see Figure 6-2), the machine pool appears as system pool identifier 2 on the Work with System Status (WRKSYSSTS) display. We recommend that you keep all Domino jobs in the *BASE pool, unless you decide to isolate the storage for one or more Domino servers in one of the shared pools *SHRPOOL1 to *SHRPOOL60.

General shared pools (*SPOOL, *INTERACT, *SHRPOOLxx)

There can be up to 62 shared memory pools in OS/400. As the name implies, they can be shared by two or more subsystems. However, to enable automatic performance adjustment for a pool used by a Domino server, it may make sense to assign one of the shared pools to a single Domino subsystem.

The size and activity level of a shared pool, as well as the *expert cache* behavior (*paging option*), can be changed with the Change Shared Pool (CHGSHRPOOL) or Work with Shared Storage Pools (WRKSHRPOOL) command.

After you define the size and activity level for one of the shared pools, you use the Change Subsystem Description (CHGSBSD) command to assign the pool to the subsystem. For example, to assign *SHRPOOL3 to the subsystem DOMINO02, enter:

```
CHGSBSD SBSD(QUSRNOTES/DOMINO02) POOLS((1 *SHRPOOL3))
```

To assign the *BASE pool to the subsystem again, you would use the following command:

```
CHGSBSD SBSD(QUSRNOTES/DOMINO02) POOLS((1 *BASE))
```

The following rules apply to Domino subsystems:

- ▶ The subsystem description for each Domino server resides in library QUSRNOTES.
- ▶ The name of the subsystem description is, by default, DOMINOxx or the name you assigned to it when you configured the Domino server. You can find out about the existing subsystem names for Domino by using the Work with Domino Servers (WRKDOMSVR) command.
- ▶ In general, each subsystem can have up to 10 memory pools assigned to it. However, the Domino tasks will only use the first pool in the subsystem description. Therefore don't assign more than one pool to each subsystem.
- ▶ The Change Subsystem Description (CHGSBSD) command can be used while the subsystem is active. However, if you assign a new pool, it will only be used when new jobs are started, which is normally only the case when the server is started. Using the Domino console command **restart server** is *not* sufficient, because the SERVER job remains active in the same pool as before and so do all other tasks submitted by SERVER in order to complete the restart.
- ▶ The shared pools *SPOOL and *INTERACT should not be used for Domino subsystems, since they are predefined for the spool printer (QSPool) and the interactive subsystem (QINTER). QPFRADJ uses some special assumptions.

Private memory pools

Private pools can only be used by a single subsystem and are *not* automatically tuned by QPFRADJ. Therefore, we do not recommend using a private pool for a Domino server, unless you have a good reason for manually tuning a single Domino server.

For more information about creating memory pools, see the iSeries Information Center (<http://publib.boulder.ibm.com/pubs/html/as400/infocenter.htm>) or *OS/400 Work Management for Version 4 Release 4*, SC41-5306.

6.4.1 Using expert cache

You can set up the system to dynamically adjust a memory pool. Expert cache uses a storage management tuner, which runs independently of the system dynamic tuner, to examine overall paging characteristics and history of the pool. There is a setting for each memory pool that controls how aggressive OS/400 storage management is about paging data into memory. This setting is called *expert cache*.

When expert cache is turned on for a memory pool, if objects are referred to sequentially, the system brings large blocks of data into memory and delays writing changes back to auxiliary storage. This reduces the number of I/O operations used by the job. It also reduces the contention for disk drives, which in turn, reduces the time that jobs wait on I/O requests. If objects are referred to randomly, the system does not bring in large blocks of data because it would not reduce the number of I/O operations.

One advantage of using expert cache (*CALC) is that the system dynamically determines which objects should have larger blocks of data brought into main storage. This is based on how frequently the object is accessed. If the object is no longer accessed heavily, the system automatically makes the storage available for other objects that are accessed. If the newly accessed objects become heavily accessed, the objects have larger blocks of data placed in main storage.

Expert cache was created because of the need for batch jobs to page large amounts of data into memory and then process it. It was not very efficient for the batch jobs to have to keep paging data into memory, so expert cache helped to alleviate this problem.

Domino can benefit from expert cache because of its batch-like processing characteristics. Because a Domino database contains the data and the design elements, it can be very helpful to use expert cache to keep the Domino database paged into memory.

Setting expert cache

There are two paging options associated with each shared memory pool, *FIXED and *CALC. By default, all pools have the option *FIXED specified. To turn on expert cache, change the paging option for the pool to *CALC using either the Work with System Status (WRKSYSSTS) or the Work with Shared Storage Pools (WRKSHRPOOL) command.

- ▶ ***FIXED:** The system limits the amount of memory used by threads that are running in the memory pool. The system transfers data from auxiliary storage and frequently writes changed data back to auxiliary storage. When *FIXED is specified for a pool during performance data collection, the Storage Pool Activity section of the Performance Tools Component Report shows a value of 0 against the expert cache value for that pool.
- ▶ ***CALC:** The system automatically determines the best approach for handling data in the memory pool. If many threads are running in a small memory pool, the system limits the amount of memory used by each thread. If the pool for those threads has enough memory, the system determines (object by object) how much data to bring into the pool.

When *CALC is specified for a pool during a performance monitor data collection, the Storage Pool Activity section of the Performance Tools Component Report shows a value of 3 for the expert cache value for that pool.

As with any performance advice, this does not mean that expert cache is recommended for every Domino environment. Expert cache can help in environments when a single Domino database is accessed very frequently and by many different users. A Domino help desk application is a good example. This database will be heavily used by the help desk staff, not only for the function the application provides, but also for the large quantities of data stored in the Domino database.

Note: If you tune using *CALC and the system ends abnormally, the recovery time could be longer than the recovery time would be with fixed paging.

6.4.2 Activity levels of memory pools

Memory pool activity levels allow for efficient use of system resources by allowing more than one thread to be active at the same time. The activity level of a pool is the *maximum* number of threads that can be active at the same time in a memory pool. The system manages the control of this level. Often during processing in a thread, a program waits for a system resource or a response from a workstation user. During such waits, a thread gives up its use of the memory pool so that another thread that is ready to be processed can take its place.

When more threads are started than can run at the same time because of the activity level controls, the excess threads have to wait to use the processing unit (normally this wait is very short). The memory pool activity level lets you limit the amount of main storage contention in the various memory pools for your subsystems.

For the memory pool activity level, the number of threads running (or active threads) refers to the number of threads that have an activity level. That is, the threads are actually running or they may be waiting for a disk I/O operation. In this sense, active threads do not refer to threads that are waiting for input, a message, a device to be allocated, or a file to be opened. Active threads do not refer to threads that are ineligible (threads that are ready to run, but the memory pool activity level is at its maximum).

A rule of thumb for the activity level factor used for the Domino subsystem is to have a value large enough to support all the threads running on the subsystem. Use the WRKACTJOB command to find the total number of threads running in the Domino subsystem or better still use option 9 in the Work with Domino Servers (WRKDOMSVR) panel. Option 9 in WRKDOMSVR is preferable over WRKACTJOB, because it shows only a subset of all jobs and, therefore, causes less overhead. From there, you only have to press F11 twice to see the number of threads for each job (Domino task) or use option 12 to see all threads of an individual job. Most Domino tasks are multi-threaded. Examples of multi-threaded tasks include Server, Event, and AMgr.

During previous testing, we found that an inadequate number of activity levels in the main memory pool can cause time-outs for the threads in the server job, which in turn, could cause the server application program/job itself, such as Domino, to crash. This is true for OS/400 V4R4 in particular since the threading model changed in V4R4, requiring more activity levels to be required.

We also found that an insufficient number of activity levels causes the server task to end abnormally and then automatically start again, which causes a huge performance impact. These problems are reported in the QSYSOPR message queue with messages CPF0908 and CPF0909.

6.4.3 Faulting rates

Keep the page faulting rate in machine pool, memory pool 1, as low as possible because this pool contains objects that are used system-wide. One way to affect the paging in the machine pool is to adjust the size of the pool.

The automatic performance adjustor built into OS/400, QPFRADJ, calculates an acceptable level of faulting based on the number of active threads in a pool. Even if QPFRADJ = '0' (no tuning), you can use the same calculations to determine acceptable levels of faulting in your storage pools. QPFRADJ uses the faulting levels in Table 6-1.

Table 6-1 QPFRADJ faulting levels

Pool	Minimum faults	Faults per thread	Maximum faults
*MACHINE	10.00	.00	10.00
*BASE	10.00	2.00	100
Interactive	5.00	.50	200
Spool	5.00	1.00	100
Batch	10.00	2.00	100

QPFRADJ assumes:

- ▶ *INTERACT has only interactive jobs.
- ▶ *SPOOL has only spool jobs.
- ▶ *SHRPOOL1-60 has only batch jobs.

Here are some examples to illustrate this:

- ▶ In an interactive pool, if there are 40 active threads, an acceptable faulting rate would be:
 $5 + (40 \times .50) = 25$ faults per second
- ▶ If there are 20 batch threads active in a pool, an acceptable faulting rate would be:
 $10 + (20 \times 2.00) = 50$ faults per second

- If the calculation is greater than Maximum Faults, then use the Maximum Faults value.

Notes:

- The interactive guideline of 0.50 is based on the assumption that the user enters a transaction every 20 seconds and 10 faults per transaction is acceptable level. (10 faults per transaction per thread ÷ 20 seconds per transaction = 0.5 faults per thread.)
- The batch guideline is based on the assumption that a page fault takes 0.010 seconds. For one job, the acceptable level is 12 faults per second. Therefore, the batch thread would spend 12% of its time page faulting ($.010 \times 12 \times 100\%$). This should be an acceptable level for most systems.

Here are some general notes on sizing pools, activity rates, and faulting:

- The rule for setting the machine pool size is to set it to two times the number shown in the Reserved Size column.
- Notice the sum of Non Database (Non-DB) faults in all of the storage pools. The Non-DB faults include program code, thread work areas, and variables, among other structures. To affect the faulting rate in the pool (except the machine pool), change both the size and the activity level of the pool. You can use one of two methods to decrease the amount of page faults:
 - Reduce the activity levels per pool to provide more memory per activity level.
 - Increase the size of the storage pool to add more storage available per activity level.

You can use one of two methods to increase the amount of faults:

- Increase the number of activity levels per pool.
- Decrease the size of the storage pool.
- Notice the sum of the database faults in all of the storage pools. Remember that a system with no non-database faults is a dead system.

Basically, a fault is an order to acquire data from a disk and transfer it to main storage to allow changes to the data. Technically speaking, a page fault is a program notification that occurs when a page that is marked as not being in main storage is referred to by an active program/job/task.

To increase the number of activity levels, simply specify a new value for pool size on the WRKSYSSTS screen (see Figure 6-3 on page 175).

- Notice the thread state transitions. When the pool size and activity level settings are in balance with each other, the ratio of columns (from left to right) should be ten to one (10:1). Usually, when the pool size and activity level settings are correct for the workload, the transition rates fall within the guidelines.

Wait-to-Ineligible transitions need not be zero all the time. When there is a momentary period of heavy usage, it may be better to let the jobs become ineligible to avoid excessive page fault rates or thrashing.

- Increase the number of activity levels to reduce the rate of Wait-to-Ineligible transitions. For pools including interactive like threads, the Wait->Ineligible should be small (less than 10% of Active->Wait). If you see any Active->Ineligible, increase the activity level (MAXACT) by 5 to 10 until the Active>Ineligible is 0. After you increase the MAXACT value, press PF10 to reset the statistics. Do not use PF5 to refresh. You should wait at least one minute between refreshes.
- Keep the time frame of the observation period between five and 30 minutes. If the observation period is less than five minutes, the occasional peak loads tend to distract the rates of both faults and pages. If the time period is over 30 minutes, the important data may be lost because the counters holding the data may get wrapped.

Note: Remember that the purpose in tuning is to process the workload as smoothly as possible without overusing any of the system resources.

6.5 Choosing which processor priority to use

The term “interactive” has a very specific meaning in OS/400. It refers to 5250 (or 3270) green-screen based processing with some strong performance implications. Each processor has an Interactive Commercial Processing Workload (Interactive CPW) rating that provides an indication of how much interactive work can be accommodated.

Although we as Domino users would consider much of Domino's Mail and Applications to be “interactive”, in OS/400 terms the tasks that support this Domino work are actually implemented as batch immediate jobs.

When looking at changing or setting priorities, Domino has tasks with different behavior patterns; some such as the SERVER task, behave more like interactive jobs:

- ▶ They become active for a relative short period of time, typically for some seconds or less.
- ▶ Then they wait for the user's next request, usually longer than a minute.

ADMINP, AMGR, and UPDATE perform in a more batch like manner, similar to traditional batch jobs:

- ▶ They run constantly for a long time (minutes or even hours).
- ▶ They only pause when they have to wait for disk I/O or at time-slice end if another job/thread with higher priority is ready to run.
- ▶ The number of jobs/threads with higher priority (and not in long wait) is higher than the activity level for the pool.

The run priority of jobs on the iSeries is normally set based on the type of work that is being performed:

- ▶ Interactive jobs started on a 5250 (or 3270) terminal run at priority 20.
- ▶ Traditional batch jobs that are submitted to run in the background run at priority 50 by default.

The higher priority of interactive jobs ensures that requests from a user are honored immediately by possibly interrupting a batch job running with lower priority (50). Since interactive usually only spend a short time actively processing, control will be soon returned to the batch job or jobs.

All Domino jobs on the iSeries server run as batch immediate jobs and all have a run priority of 20. That is, even Domino tasks with a “batch like” behavior compete directly with other tasks or interactive jobs, which need to respond quickly. In most cases, this does *not* pose any kind of problem.

However, we have seen situations where a huge portion of the total CPU utilization was consumed by background tasks like UPDATE or AMGR. The fact that overall CPU utilization raises very high does not necessarily mean it causes problems, but when the majority of CPU resources is consumed by high priority jobs, the response times may increase.

So, if the response times for the Notes clients (and traditional interactive applications, if any) raises exceptionally high during times when you observe high CPU utilization caused by jobs like UPDATE, AMGR, it *may* help if you reduce the priority of those tasks. Note, however, you should not overestimate.

You may find out which jobs are consuming most CPU resources by using the Work with System Activity (WRKSYSACT) command, described in 4.10.4, “Work with System Activity (WRKSYSACT) command” on page 71, or the platform dependent statistics in statrep.nsf (see 5.3, “Combining performance data from Domino and iSeries” on page 152).

6.5.1 Changing the run priority of Domino tasks

With Domino R5, you can change the priority (or other runtime characteristics like time slice) of Domino jobs permanently. Prior to R5, any changes to the runtime priority of a Domino job only lasted for the life of the job. If the Domino subsystem was stopped and then restarted, the priority of the individual jobs that had been changed were reset back to the original runtime priority.

With R5, you can permanently enable a Domino job to run at a different priority than the default of priority 20. This is a two step process:

1. Create a class that defines the specified run priority. Use the Create Class (CRTCLS) command for this step. The class must be created in the library QUSRNOTES. Here is an example of creating a class named `updatecls` that will have a runtime priority of 30:

```
crtcls cls(qusrnotes/updatecls) runpty(30)
crtcls cls(qusrnotes/mailcls) runpty(28)
crtcls cls(qusrnotes/amgrcls) runpty(25)
```

2. Tell the Domino server which tasks should use this class. Domino for iSeries looks for this information in the directory `/qibm/userdata/lotus/notes/domino_classes`. You can use the Edit File Utility (EDTF) to create this file if it doesn't already exist.

The contents of this file should be `SERVER`, the name of the Domino server; `CLASS`, the name of the class you created in step #1 above; and `TASKS`, a list of the Domino jobs (or tasks) that you want to use for the class you created to obtain a different runtime priority. An example follows:

```
SERVER=Mail01
CLASS=updatecls
TASKS=update
CLASS=mailcls
TASKS=router,smtp
SERVER=App01
CLASS=amgrcls
TASKS=Amgr
```

Each time the update job is started for Domino server Mail01, it obtains the runtime priority that was defined in class `updatecls`, with a runtime priority of 30. Notice that the Domino tasks router and SMTP that which are also associated with Domino server Mail01 will have yet a different runtime priority, the priority associated with class `mailcls` (28).

The `/qibm/userdata/lotus/notes/domino_classes` file is referenced for all Domino servers on a given iSeries server. To change the runtime priority of a task in a different Domino server, specify the server with the keyword `SERVER` and then reference the class and tasks that you want to use the new runtime priority. Here the AMgr task in the App01 Domino server will adopt the run priority of the `amgrcls` class, which is 25.

Be careful when adjusting the run priority of Domino tasks. If you do not fully understand the impact of the changes you are making, you may cause other performance problems or cause the application to fail. For example, if you change the run priority of the router task to a lower run priority (30 as an example), mail may not be routed in a timely fashion and the server can get back logged.

Notes:

- ▶ All Domino jobs/tasks can have their runtime priority changed with the exception of the QNNINSTS job. That can be changed together with the default for all other task by changing the class with the same name as the domino subsystem (for example DOMINO01) in library QUSRNOTES.
- ▶ OS/400 Version 4 Release 3 (V4R3) or higher is required for this support.

6.6 DASD: System ASP, user ASPs, and independent ASPs

Typically on iSeries all disk (DASD) is shared by all jobs, using (system ASP 0). It is possible to limit disk space by dividing out some of the main DASD storage at a physical disk level into 1 to 32 auxiliary storage pools (ASP) composed of one or more disks in each user ASP.

In V5R1, the independent ASP (IASP) allows the definition of up to 67 IASPs. The IASP allows a set of disks to be moved to a different iSeries server in the case of a planned or unplanned outage of one server.

Using ASPs is less commonly done these days with the advent of RAID, mirrored disks, and storage area network (SAN) systems. But historically separating out some ASPs for journal files was essential, so that both the journal and data were not sharing the same physical unit for performance reasons and could be recovered in the event of a disk crash.

This technique with using ASPs is particularly important to separate out the transaction logging files from the Domino data files and directories for performance reasons to minimize the movement of the disk access arm.

Figure 6-3 shows the system status display, which includes the pool activity.

12/09/01 12:41:13									
ACME01									
% CPU used :				26.2	System ASP :				106.0 G
Elapsed time :				00:00:05	% system ASP used . . . :				76.8542
Jobs in system :				776	Total aux stg :				106.0 G
% perm addresses :				.007	Current unprotect used . :				7345 M
% temp addresses :				.015	Maximum unprotect . . . :				7930 M
Sys	Pool	Reserved	Max	----	DB-----	--Non-DB---	Act-	Wait-	Act-
Pool	Size K	Size K	Act	Fault	Pages	Fault	Pages	Wait	Inel
1	668800	280884	+++++	.0	.0	.0	.0	11.8	.0
2	1741704	20	123	.0	.0	.0	.0	47.2	.0
3	47184	0	9	.0	.0	.0	.0	.0	.0
4	1260104	0	17	.0	.0	.9	1.9	23.6	.0
5	1000800	316	1775	.0	.0	.0	.0	3139	.0
									Bottom
====> _____									
F21=Select assistance level									

Figure 6-3 Work with System Status display

6.7 Unleashing iSeries work management on Domino servers

The way Domino is implemented on the iSeries is to have each Domino server be its own subsystem. There are various aspects to keep in mind when configuring a Domino server that can impact the performance. Let's address some of these areas.

6.7.1 Configuring a Domino server

When you configure a Domino server on your iSeries server, a subsystem description is automatically created for you. In Domino R5 for iSeries, you have the choice of accepting the default name of DOMINOxx or providing a more descriptive name of your own choice for the subsystem.

You may want to consider creating a memory pool for every server instead of using the default values that point to a system pool *BASE as explained in 6.4, "Choosing which memory pool to use" on page 166. If you do, you should change the subsystem description in library QUSRNOTES, which was created during the configuration process. We found, however, that most Domino implementations are fine by leaving all Domino subsystems running out of the default *BASE memory pool.

6.7.2 Starting a Domino server

By default ("out of the box"), all the jobs running in the Domino subsystem are started with the same job class, and same job class name as the default subsystem description for Domino in library QUSRNOTES. All jobs running in a Domino subsystem have exactly the same run attributes by default. This works well in the majority of cases, but is not ideal from an iSeries work management point of view when many Domino servers are competing for the same resources, or when a Domino server is competing with interactive tasks executing on the system. Some of the Domino jobs are for batch-like functions, such as router, updater, and indexer tasks, where some jobs are for highly responsive functions, typically the threads for users NRPC (client) requests and HTTP requests. If you start seeing a problem on an "untuned" Domino for iSeries, this is one of the first areas to look.

You have two options to adjust the run parameters on a Domino subsystem:

- ▶ Use Operations Navigator or option 2 in the Work with Active Jobs (WRKACTJOB) panel to change the parameters of one job (and all the threads initialized by that job) at a time on a real-time basis. This changes the run priority only for the duration of the job. Once the Domino server is ended and restarted, the run priority will change back to the default of run priority 20. For details on adjusting the run priority of individual jobs, see the discussion in the following section.
- ▶ Create a partitioned server for separate function sets that enable you to create different memory pools for these servers. With this method, you can also specify different run attributes by using different class descriptions in various Domino subsystem descriptions. For example, create a separate Domino server just for mail routing purposes.

We have customer requests to lower the priority of Domino R5 jobs, including Update, Sched, and AGMR, that are not used to service users directly. For details on changing the run priority of these and other Domino jobs/tasks, see 6.5.1, "Changing the run priority of Domino tasks" on page 174.

6.7.3 Working with Domino jobs

The Work with Active Jobs (WRKACTJOB) command measures system performance by measuring aspects such as the CPU usage and response time. As mentioned before, it may be better to use option 9 from Work with Domino Servers (WRKDOMSVR) panel to consume less CPU when looking at the resource consumption of Domino jobs.

To view the Work with Active Jobs display, type the WRKACTJOB command on any command line and press the Enter key. By pressing F21, more jobs appear on the display as shown in Figure 6-4.

Pressing F11 cycles the display between the basic status, elapsed data, and information about threads as shown in Figure 6-4.

CPU %:	97.8	Elapsed time:	00:24:01	Active jobs:	260
Opt	Subsystem/Job	User	Number	Type	CPU % Threads
	DOMINO06	QSYS	514292	SBS	.0 1
	ADMINP	QNOTES	514325	BCI	.0 1
	AMGR	QNOTES	514321	BCI	.0 1
	AMGR	QNOTES	514324	BCI	.0 3
	CALCONN	QNOTES	514331	BCI	.0 1
	DRT	QNOTES	516888	BCI	.0 1
	HTTP	QNOTES	515570	BCI	.0 48
	EVENT	QNOTES	514334	BCI	1.1 8
	QNNINSTS	QNOTES	514293	BCH	.0 1
	REPLICA	QNOTES	514307	BCI	.0 1
	REPORT	QNOTES	514335	BCI	.0 1
	ROUTER	QNOTES	514310	BCI	4.2 4
	SCHED	QNOTES	514328	BCI	.0 1
	SERVER	QNOTES	514300	BCI	15.9 257
	SERVER	QNOTES	514343	BCI	45.6 256
	SERVER	QNOTES	514344	BCI	22.9 256
UPDATE	QNOTES	514318	BCI	1.3	1
====> _____					
F21=Display instructions/keys					

Figure 6-4 Work with Active Jobs display

Place the cursor on any column and press F16 to arrange the display in a descending sequence. For example, you can look at CPU%, response time, and disk I/O.

Answer the question: “Are any Domino jobs consuming a large proportion of CPU?”

Press the F10 key. Wait at least one minute, and press the F10 key again. This ensures that you have a good sample of data for the time period during which you are experiencing decreased performance.

Move the cursor to the CPU% column. Then, press the F16 key to sort the list and move those with the highest CPU usage to the top. If any individual Domino job/task is consuming large amounts of CPU, you need to determine what application functions are causing the workload for this task. It may be that too many users are active at one time for the CPU resources in demand.

You can also use the Work with Active Jobs (WRKACTJOB) command to change the run attributes of a job. The following steps show how to change the run priority for one job:

1. Press the F11 key to see the job priority.
2. Enter option 2 to change the job run attributes. Enter the desired run priority in the RUNPTY parameter on the command line to change the priority for a job. Valid run priority entries range from 1 (highest priority) through 99 (lowest priority).

Priority ranges 1 to 10 are reserved for system tasks. You should not change the run priority of Domino tasks to a priority above run priority 20 (run priority values 1 -19). This could cause unexpected problems of your Domino environment or other applications.

It is much more typical to lower the run priority of an individual Domino job by changing the run priority value to something above 20 (a good range is 21 to 35). Remember that changing the run priority of the job will affect the run priorities of all threads within the job.

3. Press Enter for the change to take affect.

6.8 Online backup with BRMS

Online backup of Domino databases and now incremental online backup on iSeries are implemented with the IBM licensed product Backup Recovery and Media Services for AS/400 (BRMS), available for OS/400 V4R4 and V4R5 as 5769-BR1 and for V5R1 as 5722-BR1.

Writing individual Domino databases to physical devices would have unacceptable performance. Therefore, the databases are backed up in groups (5 is the default). You can change this group value by adding the following entry in the notes.ini file for the server (WRKDOMSVR for Domino or WRKLQPSVR for QuickPlace - option 13):

```
SAVDOMBRM_FILES_IN_GROUP=x  
SAVLQPBRM_FILES_IN_GROUP=x
```

Here x is the number of files to group in one BRMS package.

As you increase this number, all databases of your Domino server are backed up more quickly. However, all databases in the group (that is, the x number) will be logged during the backup with the changes backed up separately. Because all the databases in the group are logged until all databases in the group are backed up, the time during which changes to the databases can occur is increased and the size of the changes backed up will increase. When the databases are recovered, the changes to the database that occurred during the backup will be re-applied. This process takes longer as the number of changes increases.

If your server is being backed up during heavy server usage, you will want to keep this group value relatively small (3 to 7), so that fewer changes occur to the database during the backup operation. As a result, recovery of the database can occur in a reasonable amount of time.

If your server is being backed up during off hours when server use is low, you can set the group value higher (10 to 20 or even higher) to speed up the backup operation, while keeping the recovery time reasonable. The maximum group value is 120. If you are saving your data to a TSM server, then you should not set your group value higher than 90.

When to take a full backup

A complete backup must be the start point of your recovery. The more you take, the less log files need to be replayed in the event of a DB loss and subsequent recovery. The more full backups you take, the slower your backup cycle is (in time taken to backup), and the more data you need to store. BRMS has a limit of not more than 98 online incremental backups of a Domino database before you must perform a full backup. We recommend that you perform an online full backup of the entire Domino data directory at least once a week.

Server backups using BRMS/400 and how QNNINBRM works

When you use the Configure Domino Server (CFGDOMSVR) or the Change Domino Server (CHGDOMSVR) command to add incremental backup support, you specify ADLSVR(*BRMS) on the command. This causes the configure (or change) support to add an entry to the ServerTasks= line of QNNINBRM.

This QNNINBRM adding task queries if a transaction log is full, copies it, and then tells Domino, via the API, that it is now available for re-use. There isn't a specific interval it checks for. If it finds that a transaction log is available, it checks more often to see if a new one becomes available to be copied. If a log isn't available, it gradually lengthens the time between checks.

So if the system is busy and filling transaction logs at a fast rate, it checks more frequently to see if a log is full to be copied. If it slows down and logs aren't being filled as quickly, then the checks gradually become longer between checks until a log becomes full. Then it shortens the time again. The work it performs is saving the changes that have been made to the databases that are stored in the TxL logs (just keeping the changes is the "incremental" part of incremental backup).

Server backups using BRMS/400 and performance recommendations

BRMS/400 copies each completed log file to a sub-directory until the process of copying the TxL files to tape or another backup medium is completed. The safest and fastest place for these BRMS backup copies of the logs would be on another system, preferably in another building. For example, if the BRMS backup sub-directory were an NFS mount point, the files would immediately be held at a safe location.

It would in any case also be preferable if that sub-directory were not on the same ASP as the open TxL logs, for performance reasons, and because if the physical disk failed (and there was no journaling, RAID mirroring or SAN available), the logs together with their recent backup copies would both be lost. Losing the TxL disks will crash Domino so all buffered Notes data will be lost and therefore data loss on restart will be high after fixup, compact, and upcall.

Faster backups, reduced volumes, increased recoverability

IBM has shown from their own use of transaction logging that they can cut backup volume by 90% – huge cost savings. You can also recover a database to any point in time with this method – not just the 24 hour snapshot most companies have now.

Section 13.4.1, "Incremental online backup with BRMS" on page 384, offers some considerations regarding Domino transaction logging, which is necessary for BRMS online backup. For more information about Domino online backup with BRMS, refer to:

<http://ibm.com/servers/eserver/series/service/brms>

6.9 Network tuning

Since all users are connected to either a local area network (LAN) or wide area network (WAN) to your Domino servers, the performance of the network is also crucial to provide decent response times to your users.

The main items that are of concern here are the maximum transmission unit (MTU) size for the line description, send and receive buffer size, port filtering, and duplex settings. For the recommendation for each of these areas, read on.

6.9.1 Maximum Transmission Unit (MTU) Size

The Maximum Transmission Unit (MTU) Size parameter affects the actual size of the line flows. By increasing the value of this parameter, you can reduce the overall number of transmissions, and therefore, increase the potential capacity of the CPU and the IOP (input/output processor).

Similar parameters also exist on the client. The negotiated value is the minimum of the server and client (and perhaps any bridges/routers), so increase them all. The recommended setting varies depending on the communications protocol that is being used:

4 MB Token Ring = 4060
16 MB Token Ring = 16388
Ethernet 802.3 = 1492
Ethernet version 2 = 1500

To change the MTU size, complete these steps:

1. Type CFGTPC on any command line. You see the Configure TCP/IP display.
2. From the Configure TCP/IP display, select option 2 (Work with TCP/IP Routes) and press Enter. You see the Work with TCP/IP Routes display.
3. From the Work with TCP/IP Routes display, type 2 in the Opt column next to the IP address used by your Domino server. You see the Change TCP/IP Route display.
4. On the Change TCP/IP Route display, type in the recommendation from above in the Maximum transmission unit (MTU) parameter.

6.9.2 TCP/IP buffer size

Web serving performance can be increased by tuning the buffer size that is used by TCP/IP, especially when sending large amounts of data. If your network is very reliable, try increasing the buffer size from the default (8000) to 64000. If your network experiences a significant amount of collisions or congestion, you may be able to improve performance by decreasing the TCP/IP send and receive buffers. This is because it will take less time to detect a bad packet, and less data will need to be retransmitted.

To change the buffer size, follow these steps:

1. From a command line, type CFGTCP.
2. From the Configure TCP/IP display, select option 3 (Change TCP/IP Attributes).
3. On the Change TCP/IP Attributes display screen, locate the TCP receive buffer size (TCPRCVBUF) and type a new value.
4. Locate the send buffer size (TCPSNDBUF) and type a new value.
5. Press Enter.

6.9.3 Port filtering

Filtering happens on every communications line, so you want to make sure that you have the proper hardware configuration to enable this support. It has happened in the past that filtering has not been setup correctly on a system and the result was a communications performance problem.

The primary concern has been with the 2838 Ethernet card. If you have this card, check the part number and ensure that filtering is optimally configured.

If the part number is 21H9067 on the 2838 Ethernet card, then an external filter is required. The external filter cable with part number 97H7385 is recommended.

If the part number on the 2838 Ethernet card is 21H5458, or anything else, it has built in filtering, so you should not have an external filter since this double filtering can cause problems.

6.9.4 Duplex

Ethernet supports both half and full duplex. The best performance will be with FULL duplex. However, the duplex setting on the line description must match the setting on the port on the switch if the line is hooked up to a switch. Be especially careful if you set your Ethernet switch or your line description to *AUTO. In many cases, we have found where performance is severely degraded because the duplex setting did not auto-negotiate correctly. It is probably best to configure switches and line descriptions to either *FULL or *HALF explicitly.

If the line is connected directly to a 8271 Ethernet switch, the switch's port is FULL duplex. Therefore, the duplex setting on the Ethernet line description should be set to FULL.

If the line is connected to a "stackable" hub, then the duplex setting on the line should be set to HALF to match the hub setting.

6.9.5 TCPONLY for Ethernet

TCP/IP performance can be even further improved in V4R4 by setting the TCPONLY parameter to *YES on an Ethernet line description. This decreases the TCP/IP code path length by limiting the code that is loaded in the IOP for the line description. Ethernet was a focus area in improving communications performance in V4R4. To get the best TCP/IP performance, upgrade to V4R4 and take advantage of the improvements and set up your network to use full duplex.



Tuning Lotus Domino for better performance on iSeries

A Domino server is no more (or less!) than a complex application running in an environment of its own. To effectively and successfully tune such an application, you obviously have to know what the application is doing, what all the smaller parts of the application are doing, and how it all ties in together. For example, consider a case where you reduce the priority on what appears an insignificant batch-like thread. You could kill the performance or even the entire application, if, for example this was the common and necessarily single-threaded lock handler! So you must know what you are tuning if you want consistent and predictable results.

This chapter describes what parameters and configuration options within the Lotus Domino environment can be changed to influence the performance of the server. See Chapter 6, “Tuning the iSeries server for Lotus Domino” on page 161, for the influence of OS/400 configuration parameters on the performance of Lotus Domino.

This chapter provides details on:

- ▶ Tuning Domino using the notes.ini parameters
- ▶ Special considerations for Domino tasks, for example UPDATE, that often have a high impact on overall resource utilization

7.1 Basics of Domino tuning

The following sections emphasize the possibilities a Domino administrator has to optimize the workload that a Domino for iSeries server can handle. Tuning for best performance, however, starts with a good application design during the development phase. More information on application development performance for Domino can be found in the redbook *Performance Considerations for Domino Applications*, SG24-5602.

The best way to improve Domino performance is to:

1. Plan ahead.
2. Test the Domino application thoroughly on a test system and then on a pilot system.
3. Start performance monitoring from day one before the first user signs on.
4. Constantly analyze your performance data and adjust your sizing based on actual facts.
5. Implement the application on the production system.
6. Roll it out gradually, a few users at a time, constantly monitoring its performance.
7. Continue to analyze the performance data and report future trends and hardware needs.

Remember: A very obvious but important lesson about server tasks and functions is that if your organizational requirements have no need of a particular function, *do not* start and run it on your server!

It has been considered best industry practice in recent years to use a centralized server or workstation to centrally capture, collect, and report the statistics of a group of servers. If you have this type of “background” processing on one of your iSeries machines, always set the run priority and time slice values relatively low on that subsystem description. This keeps these background activities in the background and does not disturb the productive work of the system.

7.2 Relating Domino tasks to iSeries jobs

A Domino subsystem contains a number of jobs. Each supports a different basic Domino task. The following list explains the OS/400 job names and their corresponding tasks on the Domino side:

ADMINP	The major Domino scheduler job running batch-like updates
AMGR	Domino Agent Manager; takes care of pre-defined agents
CALCONN	Connects the Domino calendar to other types of calendars
EVENT	Domino Event Handler
QNNINSTS	Monitors and manages automatic recovery (No corresponding task)
REPLICA	Domino database Replicator Task
REPORT	Domino Report Generator (adding data to STATREP.NSF)
ROUTER	Domino Mail Routing Task
SCHED	Domino Scheduler Task
SERVER	The main Domino Server Task
UPDATE	UPDATE task for Domino database indexes and views

For a fully detailed description of these functions, see Chapter 8, “Understanding the Domino server jobs” on page 229. After you determine what is using the excessive CPU or any other system resource, the Domino administrator and OS/400 operator should work together to reduce or remove the problem.

7.3 Domino parameters that affect performance

The major tuning tool for Domino is the notes.ini parameters found both in the notes.ini file and in the configuration documents in the Domino Directory (names.nsf formerly called the Name and Address book or Public Address Book). Many of these settings drastically affect Domino performance. They are sorted by function in tables starting with Appendix C, “Important notes.ini parameters” on page 435. For more detailed information on notes.ini parameters, see:

- ▶ Notes Administration help (installed on your Domino server as file help5_admin.nsf)
- ▶ Search Notes.Net Ask Professor INI: <http://www.notes.net/today.nsf/profini?OpenView>
- ▶ The Prominic.net site has a useful lookup and descriptive list: <http://domino-8.prominic.com/>
- ▶ Chapter 42 and Appendix E in *Lotus Domino 5—Administering the Domino System*, which comes in two volumes with your Lotus Domino server.

7.4 Domino Database indexing: Controlling the UPDATE task

One of the most common, preventable causes of poor response time and performance of Domino servers is excessive and unnecessary activity of the Domino UPDATE task. This task is responsible for updating and rebuilding the indexes of Domino database views. It is designed to run in the background and is intended to improve response time and performance by ensuring that when a user opens a database view, the user does not have to wait for it to be indexed. Ironically, it often turns out to have the opposite effect.

This section explains how the UPDATE task is supposed to work. It looks at how Domino administrators and designers can monitor and control it to improve server performance. It also explains the potential impact to performance when the UPDATE task is running excessively. It provides a hint of the kind of improvements you might see by correcting the problem.

7.4.1 Components of the Domino database indexer

Lotus Technote #167013 provides a great deal of information regarding the Domino Indexer and the UPDATE task. This section includes a few excerpts that are relevant to the topics in this book. However, you should read the entire Technote if you want to understand what happens under the covers. You can find the Technote by searching for 167013 on the Lotus Support Web site at: <http://www.support.lotus.com>

The *Indexer* is composed of three components:

- ▶ The UPDATE task
- ▶ The UPDALL task
- ▶ The Notes Indexing Facility (NIF) Subsystem

The Indexer, as a whole, is responsible for keeping the active views and the full text indexes current, within the databases. It processes any requests for changes to documents within a database, so that the active view collections and the full text indexes display the most recent modifications and current information.

The UPDATE task

UPDATE is a Domino server task that should run at all times (under normal conditions; this is explained later). You can load the UPDATE task in one of two ways:

- Automatically, by specifying it in the “ServerTasks=” line in the notes.ini file. This is the default when you configure a Domino server. Here's an example of the “ServerTasks=” line:

```
ServerTasks=Replica,Router,Update,Stats,AMgr,AdminP,Sched,....
```

- Manually, from the Domino console, using this command:

```
load update
```

The UPDATE task works *continuously* from a queue called *\$UpdateQueue*. When a change occurs in a database (such as deletions, additions, or edits), a corresponding request is entered into the UPDATE queue. UPDATE checks the queue every 5 seconds for any new requests that have been deposited and takes the requests from the queue on a first-come, first-served basis.

\$UpdateQueue is a hard-coded queue that has a maximum capacity of 500 requests. These requests include updates to view indexes as well as full text indexes. When a request is placed into the queue, the actual names of the databases and paths to the databases are stored in the queue, not the view names themselves. No published method is available to display the actual contents of the \$Update queue.

To see if UPDATE is running and what it is doing, you can use the **show tasks** console command. From the Domino console, it is displayed as the Indexer task as shown in Figure 7-1.

```

Work with Domino Console
Server: FMS170

Previous subcommands and messages:
Statistic Collector Idle
DECS Server Idle
Event Monitor Idle
Calendar Connector Idle
Schedule Manager Idle
Admin Process Idle
Agent Manager Executive '1': Idle
Agent Manager Idle
Indexer Full text indexing documents in m_dir/MSS_TS_C.nsf
Router Idle
Replicator Idle
Billing Idle
QNNINBRM BRMS Log Manager Idle

Enter a Domino subcommand.
===> _____

F3=Exit F5=Refresh F6=Print F9=Retrieve
F17=Top F18=Bottom F21=Command line

```

Figure 7-1 The Indexer task at the Domino console

However, when you use OS/400 functions (for example, the Work with Active Jobs (WRKACTJOB) command as shown in Figure 7-2 or using the Operations Navigator), it appears as an OS/400 job called *UPDATE*.

```

Work with Active Jobs
FRA820
08.11.01 17:37:28
CPU %: 4,7 Elapsed time: 00:30:55 Active jobs: 209

Type options, press Enter.
2=Change 3=Hold 4=End 5=Work with 6=Release 7=Display message
8=Work with spooled files 13=Disconnect ...

Opt Subsystem/Job User Type CPU % Function Status
— UPDATE QNOTES BCI 0,0 PGM-UPDATE SELW

Parameters or command
===>
F3=Exit F5=Refresh F7=Find F10=Restart statistics
F11=Display elapsed data F12=Cancel F23=More options F24=More keys
Bottom

```

Figure 7-2 The Indexer task on the Work with Active Jobs panel

The UPDALL task

The UPDALL task is a single instance of the UPDATE task. It does not operate off a queue like UPDATE, but accomplishes the same purpose in a different way. When the UPDALL task runs, it processes every Notes database, refreshing the views or updating the full text indexes for each database. The UPDALL task starts in one of two ways:

- ▶ Automatically, at a scheduled time each day. By default, the UPDALL task is run at 2:00 AM. It is defined in the ServerTasksAt2 line in the notes.ini file as follows:

```
ServerTasksAt2=Upda11
```

- ▶ Manually, from the Domino console. When you run the UPDALL task from the console command line, you can use various switches to control it. For example, you can invoke UPDALL to rebuild only the full text index on a single database.

The UPDALL task performs additional functions beyond what the UPDATE task does. In the design of a view, you can specify, in *View Properties*, the frequency with which a view index is discarded:

- ▶ Never
- ▶ After each use
- ▶ If inactive for a specified number of days

It is the responsibility of the UPDALL task to remove the view index if a discard option is specified. In other words, if the Discard Index option is selected for a view, the index is not actually discarded immediately. Rather, the index is removed the next time the UPDALL task runs. For example, a view has a Discard View option of “After each use”. If you exit from a view at 1:00 p.m., the view will not be removed until UPDALL is run (usually done at 2:00 a.m.).

Notes Indexing Facility

The *Notes Indexing Facility* (NIF) is made up of a multitude of functions within Domino that allow a server to keep data notes ordered and current within a view. Specifically, the NIF performs these tasks:

- Updates Indexes
- Opens and closes view collections/view indexes
- Locates Index entries

The majority of these requests are made by the server when users open and close databases. For example, the NIF subsystem – *not* the UPDATE task – forces the update of a view collection when a user switches between views. If a user makes a modification to a document within a view, and then switches quickly to another view and then back again to the original view, the new changes should be seen almost immediately by the user.

The rest of this section focuses on the UPDATE task.

7.4.2 UPDATE task CPU utilization in a normal environment

On a typical Domino server (one that is not experiencing performance problems due to the UPDATE task), UPDATE may run quite often, but usually for very short periods of time. Unless a database has just had several hundred or thousand documents added or changed, UPDATE does its work quickly, updates each view in the database it is indexing, and finishes in a few seconds.

With small databases, even on older, slower hardware, doing an incremental update of a view, in which only a few or no documents have changed, takes less than a second. If you observe CPU utilization when this is taking place, you may see the utilization rise to 90% or more (on a single CPU), but only for a few seconds while UPDATE traverses all the views in a database.

With very large databases (as long as the number of new or changed documents is small), the time to do an incremental update of a view still remains short, usually only a second or maybe a few seconds.

In any case, with small or large databases, or fast or slow servers, the time it takes to do an incremental update of the views in a given database will be similar for all views in that database as long as the number of new or changed documents is small.

This activity (incremental updates of view indexes) does not usually cause much of a performance problem for servers. Although the utilization is high, it is brief. Its effect is hardly noticeable. Also, although views are also “locked” and, therefore, inaccessible when they are being updated. No one usually notices because it normally takes, at most, a few seconds.

You should also be aware that the time to do an incremental update of a view, in which few or no documents have changed, depends more on the number of documents in the *database* than the number of documents in the *view*. This may sound contrary to what you have read, but it is not. When the number of documents to be indexed is low or zero, the time to update is really mostly just overhead and depends on the database size. However, if you consider the time it takes to completely *build* a view, then it is the number of documents in the *view* that matters more than the actual number in the *database*.

7.4.3 The UPDATE task and CPU utilization in a problem environment

Now, if you consider a server in which the UPDATE task is observed to be running almost *all the time* and in which the CPU utilization is consistently at or near 100%, usually something else is going on. This does not necessarily mean, the UPDATE task permanently consumes over 90% of all processor resources. There can also be cases where the overall utilization approaches 100% and the UPDATE task accounts for the *major* part of it.

A high total CPU utilization does not necessarily cause problems on an iSeries server. However, if high priority jobs, such as Domino tasks or interactive (5250) jobs consume a major part of resources (including disk utilization), you can see the following results:

- ▶ All jobs with same priority *may* see longer response times.
- ▶ Jobs with lower priority, for example traditional batch jobs with lower priority (that is, the priority value is higher than 20), will drastically slow down or even stop working.

Several reasons may cause the UPDATE task to use up too many processing cycles or even monopolize the CPU:

- ▶ The total *number of views* in databases with many documents, to be maintained by the UPDATE task, is extremely high.
- ▶ *Full text indexes* are created frequently.
- ▶ Many views (for big databases) have to be *completely rebuilt from scratch*.
- ▶ Many views contain *time-dependent selection formulas*.

Too many views or full text indexes

Since views and the capability of full text search represent the full power of many Domino applications, it is in most cases not an option to reduce any of them. If these two reasons turn out to be the cause of excessive resource consumption, probably the only choice is to invest in more and faster hardware.

Views have to be completely rebuilt from scratch

When the UPDATE task is called upon to completely rebuild a view from scratch, it is also very CPU-intensive. So far, this is the bad news. The good news is that this should normally not happen. UPDATE should *almost never* need to completely rebuild a view. Most of the time, this is redundant and a waste of CPU resources.

The only circumstances in which UPDATE should rebuild views from scratch are:

- ▶ All the documents in the view or database have changed or are new. This might occur because:
 - An agent or external program has updated them.
 - A Lotus Enterprise Integrator (LEI) Direct Transfer Activity deletes and refreshes all the documents in a database.

This should be avoided whenever possible. Sometimes other techniques or tools can be used, such as using LEI Replication Activities, which have a smaller impact.
- ▶ The database is new. If a new replica is created on a server, and some of the views have the proper settings, UPDATE builds the indexes for those views.
- ▶ The design of the view has changed. If the view has the proper settings and previously existed, changing the design causes the whole view index to be rebuilt.

As you might surmise, the circumstances in the second and third bullets should not be common occurrences.

Time dependent views

If the examples above are not the cause of views having to be rebuilt, then the likely cause is the existence of one or more views with time-dependent selection formulas in which the View Refresh Index options specify that the views should be automatically updated. The UPDATE task has no business rebuilding these kinds of views. Why? Because they are *always* out of date.

Whenever this kind of view is opened by a user, the view index is built from scratch, on the fly, for that user by the Notes Indexing Facility. It doesn't matter if UPDATE built it 5 minutes ago. It will be rebuilt again. And if another user opens it, it will be rebuilt again. Since it is always rebuilt when opened by a user (or by an application, for that matter), it makes no sense for UPDATE to rebuild it too.

The wisdom of having views with time-dependent functions in their selection formulas is not discussed here, except to say that there are legitimate reasons to do this and that it is feasible, if managed correctly. The following section explains how.

7.4.4 Affecting the UPDATE task behavior through the view properties

Before solutions to the CPU utilization problem are discussed, here is a review of the rules that the UPDATE task follows. The UPDATE task will update or rebuild a view index if and only if either the following are true:

- ▶ The Refresh Index option for a view is set to "Automatic"
- ▶ The view exists already

You can use combinations of the *Refresh Index* and *Discard Index* options within the view properties (see Figure 7-10 on page 197) to control whether UPDATE rebuilds a view.

For example, consider a view that is used rarely, say once a month, which displays only those records created in the last 30 days. You might specify the Refresh option as "Auto, after first use" so that the user always sees the most recent information, yet specify the Discard option as "After each use". Therefore, according to the second rule above, UPDATE always ignores the view. You also save the disk space that the view index might otherwise have used.

The solution to the CPU Utilization problem is to use these View Refresh Index and Discard Index Options properly.

Setting the view properties to improve performance

Here are some recommendations for how you might set these index options in the view properties, depending on what you want users to see and how often the view is used.

Any view that has a view selection formula with a time-dependent function in it

Time-dependent functions, such as @Now or @Today, are sometimes used in selection formulas for views for example to show all documents that have been created today or that are older than a certain number of days. For an example, see Figure 7-9 on page 197. These kind of views should have its Refresh Index option set to either:

- ▶ Auto, after first use
- ▶ Manual
- ▶ Auto, at most every...

It should *never* be set to "Automatic."

This controls how often the view is updated by users (that is through the NIF; see "Notes Indexing Facility" on page 187) who open the database. However, it does not control how the UPDATE task handles it. It is important to understand that even though it makes no sense for UPDATE to rebuild a view that has a time-dependent selection formula, if the view exists, UPDATE will rebuild it. A suggestion to change this behavior has been submitted to development for consideration in the next major release of Domino.

The index of a view of this type is *always* out of date. The setting of “Automatic” tells UPDATE to always rebuild the view, even if it does not exist. This setting would defeat the purpose of the Discard Index option, and since the view index will be obsolete as soon as UPDATE is finished, this should never be done. It is a waste of CPU.

Use *Auto, after first use* if you want users to always see the latest information in the view and if the contents of the view might change between the times that UPDALL or UPDATE is run, in other words, nightly, or if all changes come as the result of replication. (UPDATE will run after a replication.)

Select *Auto, at most every nn hours* if the contents of the view changes more frequently than nightly and if you need users to see the most recent data very soon after it changes.

Select *Manual* when users *do not need* to see the most recent changes unless they specifically ask for it. In reality, UPDATE or UPDALL make sure that the view is no more stale than one day, assuming you run UPDALL on the server nightly.

Time-dependent function and index refresh option set to ‘Auto, after first use’

Any view that has a view selection formula with a time-dependent function in it, such as @Now or @Today, and having its index refresh option set to “Auto, after first use”, should have its discard index option set to *After each use* (see the example in Figure 7-11 on page 198). This is necessary to ensure that the UPDATE tasks sees no index on the view and ignores it.

Time-dependent function and index refresh option set to ‘Manual’

Any view that has a view selection formula with a time-dependent function in it, such as @Now or @Today, and having its index refresh option set to “Manual”, should have the discard index option set to *Never*. The refresh option of “Manual” allows the view index to be reused by other users.

The view index is refreshed manually by pressing Shift+F9. This view always opens very quickly. However the user who presses Shift+F9 to refresh the view may have a long wait as it is being updated. Optionally, the discard option may be set to “If inactive for xx days”. But this would only be done if there was a need to conserve disk space used by the view.

Seldom or never used views

Views that are seldom or never used can have their discard index option set to “If inactive for xx days if there is a need to conserve disk space”. This can also shorten the time it takes UPDATE to refresh views, but the benefit depends on how many changed documents there are in the database.

For databases with lots of documents (or all of them) changing every day, this setting is useful for controlling the indexes on views that aren't used. If used, the refresh index option should be either “Manual” or “Auto, after first use”. Setting it to “Automatic” negates the affect, since UPDATE always rebuilds the view (it will never be discarded).

When should you use the ‘Automatic’ options?

The refresh index option of “Automatic” should only be used on views that:

- ▶ Do *not* have time-dependent selection formulas
- ▶ Are frequently used
- ▶ Have a Discard Index option of “Never”

Generally, “Auto, after first use” is a better choice. The difference between these two settings would only be noticed:

- ▶ The first time the view is used
- ▶ The first time it is used after being copied using the Notes client (File Database Copy)
- ▶ After having all view indexes discarded on purpose using UPDALL (unlikely except perhaps when troubleshooting a bug)

Now that you understand the problem and what to do about it, you might wonder if your server has this problem and how you might determine which views and databases, if any, might be the cause.

7.4.5 Detecting UPDATE task problems

You might wonder if there is something that might be a tip-off that a Domino server is suffering from the problem described above. CPU utilization statistics alone may not be enough of an indicator. The real tip-off is an indication that the UPDATE task is running at high CPU utilization rates *continuously*. On OS/400, you can use Management Central (a component of Operations Navigator) or the Work with System Activity (WRKSYSACT)¹ command to monitor utilization by job or by thread. If you set the refresh rate to around 15 seconds, and watch for a while, you can usually tell if UPDATE is the cause.

If UPDATE is using 60 to 70% of a single CPU *and* the total CPU utilization is at 95 to 100% for more than a minute straight, that's a tip-off. If it only runs that high for a few seconds and then drops and rises again, it is not so likely to be the result of unnecessary view index rebuilds.

If you want to know for sure, you need to analyze your Domino server's use of UPDATE.

Analyzing your Domino server

Determining whether your Domino servers have any databases with views that are being rebuilt unnecessarily is actually rather easy. You need access to the Domino Server log database (log.nsf) and add one statement to the notes.ini file on each server you want to monitor.

The basic process is summarized below. Each step is described in more detail following the summary.

1. Add the line LOG_UPDATE=2 to notes.ini.
2. Restart the UPDATE task.
3. Collect data.
4. Filter the log file (using the Domino Administrator's Analyze feature) to capture only the entries related to the UPDATE task.
5. Look for views in any database in which the time to update the view is considerably longer than to update most of the other views *in the same database*.
6. Examine the design and index options of the views you identify in step 5.

Note: Another way to approach this is to look at the Database Sizes view in the log and look at the view sizes of the largest databases. The largest views in the largest databases are good candidates for being the culprit when UPDATE is running wild. But this process is trial and error, and you still have to examine the design of the suspected views.

Let's take it one step at a time:

¹ The Work with System Activity (WRKSYSACT) command exists only on your iSeries server if the Performance Tools (5722-PT1 or 5769-PT1) are installed. As an alternative, the Work with Active Jobs (WRKACTJOB) can be used, but it has several disadvantages.

1. Add the line LOG_UPDATE=2 to notes.ini.

This entry causes the server to log information whenever the UPDATE task updates views on the server. It records a time stamp, the database name, and the view name as shown in Figure 7-3.

Applications-01/HQ/Acme
12/19/2000 11:07:33 AM -

Databases accessed: 3 Documents read: 175 Documents written: 0
12/15/2000 09:20:40 AM Opened session for Terry Smith/MFG/HQ/Acme (Release 5)
12/15/2000 09:20:42 AM Closed session for Terry Smith/MFG/HQ/Acme
Databases accessed: 3 Documents read: 0 Documents written: 0
12/15/2000 09:21:04 AM Opened session for Dave Jones/SALES/HQ/Acme (Release 5)
12/15/2000 09:22:34 AM Closed session for Dave Jones/SALES/HQ/Acme
Databases accessed: 21 Documents read: 1 Documents written: 0
12/15/2000 09:23:21 AM Opened session for Rick Spice/SALES/HQ/Acme (Release 5)
12/15/2000 09:24:10 AM Closed session for Dave Jones/SALES/HQ/Acme
Databases accessed: 4 Documents read: 0 Documents written: 0
12/15/2000 09:24:17 AM Opened session for Tom Kennedy/HR/HQ/Acme (Release 5)
12/15/2000 09:24:17 AM Closed session for Tom Kennedy/HR/HQ/Acme
~~Databases accessed: 3 Documents read: 0 Documents written: 0~~
12/15/2000 09:24:31 AM Updating tech/TimeTrak.nsf view 'Time Log\Payroll'
12/15/2000 09:24:33 AM Searching Administration Requests database.
12/15/2000 09:24:40 AM Updating tech/TimeTrak.nsf view 'Time Log\Payroll\HR No Authority'

Figure 7-3 Miscellaneous events view in LOG.NSF

The server also logs information when UPDALL runs. When you enable this logging on a server, it generates more information in the log, but the additional logging should not have a noticeable affect on server performance. It is safe to use on a production server. It is documented in the Domino 5 Administration Database (help/help5_admin.nsf) under **Reference -> NOTES.INI Settings -> Log_Update**.

2. Restart the UPDATE task. To restart the UPDATE task, issue the following command from the Domino Console:

```
TELL UPDATE QUIT
```

Wait for notification that it has ended, as shown in Figure 7-4.

```
tell update quit  
> tell update quit  
02/07/2001 02:18:27 PM Index update process shutdown
```

Figure 7-4 Ending the UPDATE task

3. When the UPDATE task has ended, issue the following command:

```
LOAD UPDATE
```

Normally, this does not affect any users because UPDATE runs in the background. The only possible side effect might be that someone encounters a long delay opening a view for the first time if that person is the first one to do so. But this would only happen if the database was new or all the documents were just added or changed. If this is a possibility, then you should schedule the restart off-shift.

4. Collect data.

It is best to collect a few hours worth of data during the busiest time of the day (or night), whenever you are experiencing the worst response times and highest CPU utilization. It is even better if you can be sure, ahead of time, that UPDATE (and not some other task or job) is responsible for the high utilization. With OS/400, you can do this easily using Management Central's performance monitor. The platform dependent statistics (see 5.1.4, "Platform-dependent statistics" on page 137) may also be very helpful.

When you have collected enough data, you can remove the entry from the notes.ini file and restart the UPDATE task.

5. Filter the log.

Use the Domino Administrator to select the log output you want to examine. From the Domino Administrator window, select the **Server** tab and then the **Analysis** tab as shown in Figure 7-5. Then click the **Analyze** button and select **Log...**.

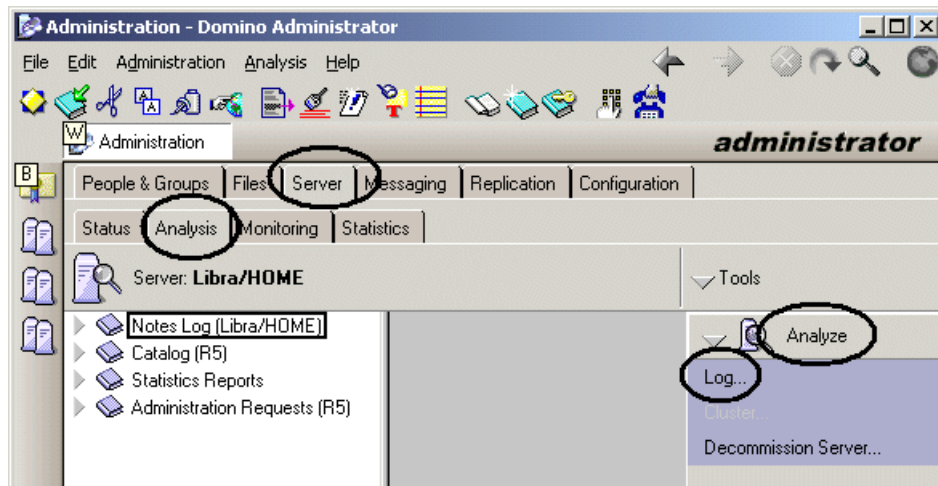


Figure 7-5 Analyzing LOG.NSF with the Domino Administrator

6. After you click Log..., you see the Server Log Analysis dialog window as shown in Figure 7-6. Enter the text Updating and set the other options as appropriate. As a result of

the analysis, a new database can be created on your client or the Domino server, or the logging entries may be added to an existing database.

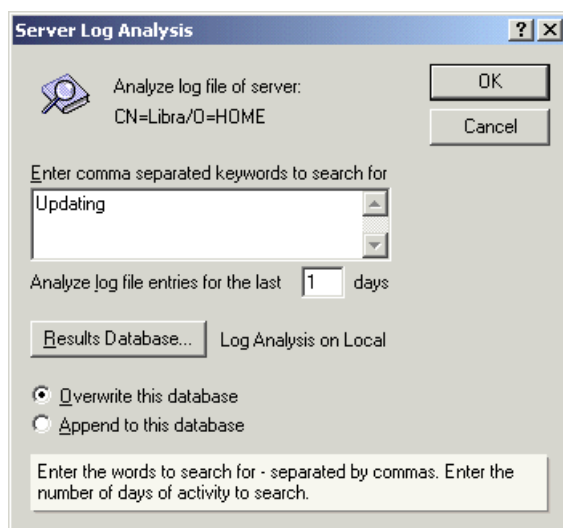


Figure 7-6 Server Log Analysis dialog

- After the analysis has been performed, the database containing the results opens automatically show a view similar to the example in Figure 7-7.

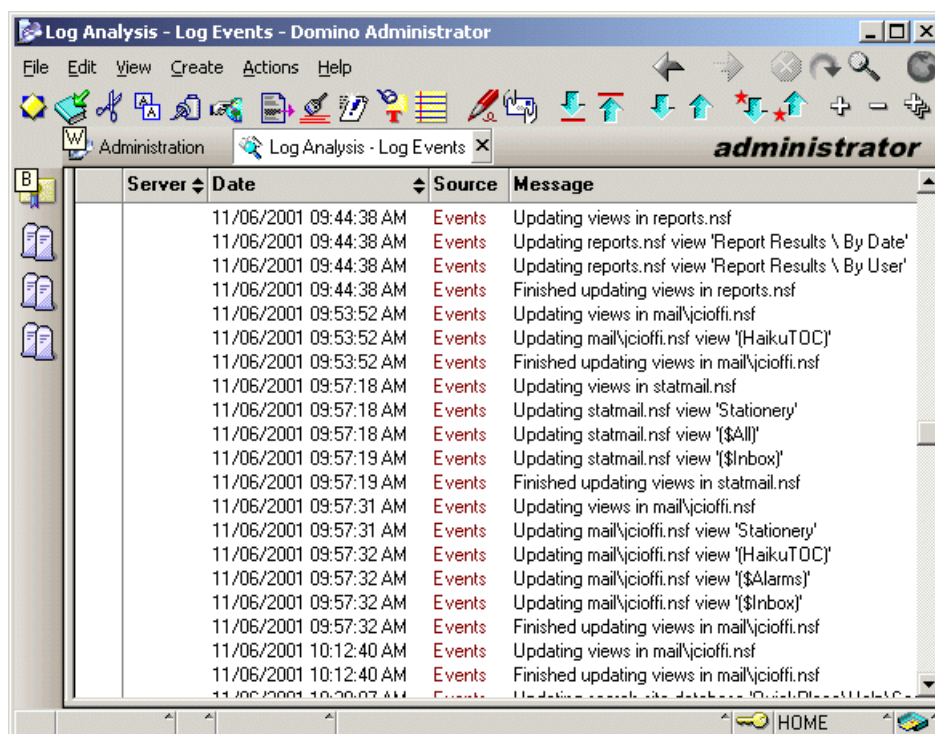


Figure 7-7 Log Analysis results

- Look for problem views.

A “problem view” meets this criterion: the time to update the view is more than twice the time to update most other views *in the same database*. With the information that you have in analysis results (Figure 7-7), you can calculate the time it takes to update each view by looking at the time stamps of the entry for that view and the subsequent entry.

In the example shown in Figure 7-8, the “All Documents” view was updated beginning at 9:10:06 and ending at 9:10:07, taking about 1 second. In fact, most of the views took about 1 or 2 seconds to be updated. However, the “Time Log\Manager Reports\Not Submitted” view started at 9:10:26 and took more than 14 minutes to be updated. This is a clear indication of a problem.

First of all, if all the documents in the database were new or changed, most or all of the other views would also take a long time to rebuild. You need to examine this view and the “Time Log\Payroll\HR No Authority” view (more than six minutes to be updated).

```
12/15/2000 09:10:06 AM Updating views in tech/TimeTrak.nsf
12/15/2000 09:10:06 AM Updating tech/TimeTrak.nsf view 'Admin\VoidedTimeLogs'
12/15/2000 09:10:06 AM Updating tech/TimeTrak.nsf view 'All Documents'
12/15/2000 09:10:07 AM Updating tech/TimeTrak.nsf view 'Calendar'
12/15/2000 09:10:07 AM Updating tech/TimeTrak.nsf view 'Error\Entry Count\By Week\By Territory\By Emp-Date'
12/15/2000 09:10:07 AM Updating tech/TimeTrak.nsf view 'MenuCodes'
12/15/2000 09:10:07 AM Updating tech/TimeTrak.nsf view 'Menus'
12/15/2000 09:10:08 AM Updating tech/TimeTrak.nsf view 'Time Log\By Actual\By Date-Emp'
12/15/2000 09:10:10 AM Updating tech/TimeTrak.nsf view 'Time Log\By Emp-Week'
12/15/2000 09:10:11 AM Updating tech/TimeTrak.nsf view 'Time Log\By Emp-Week Detail'
12/15/2000 09:10:14 AM Updating tech/TimeTrak.nsf view 'Time Log\By Manager\By Date-Emp'
12/15/2000 09:10:15 AM Updating tech/TimeTrak.nsf view 'Time Log\By Manager\By Emp-Date'
12/15/2000 09:10:17 AM Updating tech/TimeTrak.nsf view 'Time Log\By Territory\By Date-Emp'
12/15/2000 09:10:18 AM Updating tech/TimeTrak.nsf view 'Time Log\By Territory\By Emp-Date'
12/15/2000 09:10:20 AM Updating tech/TimeTrak.nsf view 'Time Log\By Week-Territory-Emp-Date'
12/15/2000 09:10:21 AM Updating tech/TimeTrak.nsf view 'Time Log\By Week\By Territory\By Date-Emp'
12/15/2000 09:10:22 AM Updating tech/TimeTrak.nsf view 'Time Log\By Week\By Territory\By Emp-Date'
12/15/2000 09:10:23 AM Updating tech/TimeTrak.nsf view 'Time Log\Customers'
12/15/2000 09:10:25 AM Updating tech/TimeTrak.nsf view 'Time Log\Lookup Start Time'
12/15/2000 09:10:26 AM Updating tech/TimeTrak.nsf view 'Time Log\Lookup'
12/15/2000 09:10:26 AM Updating tech/TimeTrak.nsf view 'Time Log\Manager Reports\Not Submitted'
12/15/2000 09:24:40 AM Updating tech/TimeTrak.nsf view 'Time Log\Payroll\HR No Authority'
12/15/2000 09:31:07 AM Updating tech/TimeTrak.nsf view 'Payroll\HR Appr'
12/15/2000 09:31:08 AM Finished updating views in tech/TimeTrak.nsf
12/15/2000 09:31:14 AM Updating views in names.nsf
12/15/2000 09:31:14 AM Updating names.nsf view 'Locations'
12/15/2000 09:31:14 AM Updating names.nsf view 'People\By Employee ID'
12/15/2000 09:31:14 AM Updating names.nsf view 'People\By Employee Standard Name'
```

Figure 7-8 Log analysis example

9. Examine the design.

Now that you have identified the views that are causing UPDATE to run excessively, you should examine the design of the views to determine if the view has a time-sensitive function in its selection formula and to see what the index refresh and discard options are. You need designer access to the databases to do this.

7.4.6 Making the changes

To illustrate how to make the changes, we assume that the view in the example above is the one we are going to examine. Follow these steps:

1. Using Domino Designer, open the database and view you want to check. Select the **View Selection** object as shown in Figure 7-9 and look at the formula. In this case, it contains the function @Today. This means that you need to change the Discard index view option.

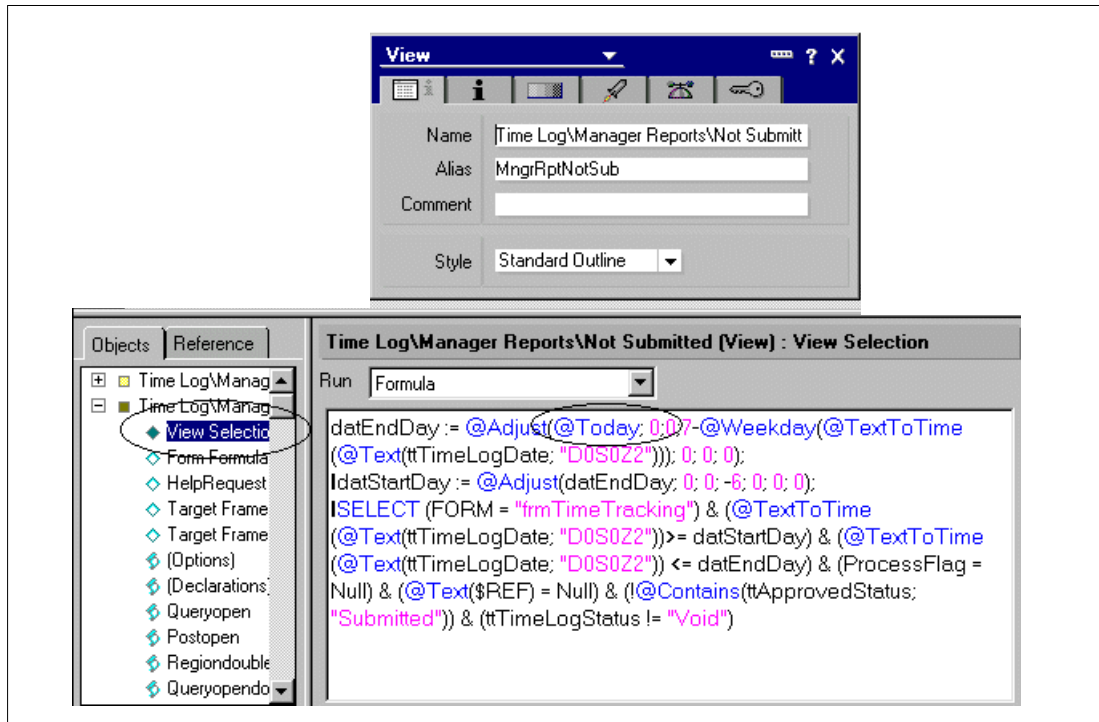


Figure 7-9 Example for a time-dependent selection formula

2. Select **Design-> View Properties**. Click the **Advanced** tab in the Properties box. You then see a display like the example in Figure 7-10.

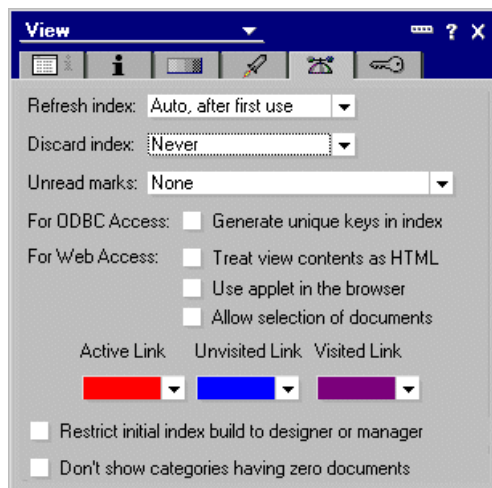


Figure 7-10 Default settings of the View properties

3. If the Refresh index option is set to “Auto, after first use”, and the Discard index is set to “Never”, change it to **After each use** as shown in Figure 7-11.

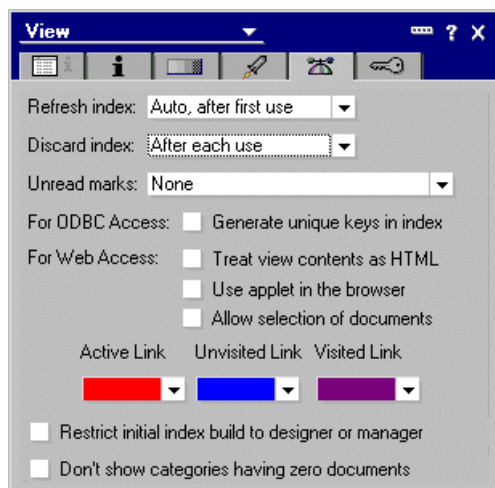


Figure 7-11 View Properties changed to ‘Discard index after each use’

Once you make the change, save the updated view.

Note: If the database you are changing is linked to a design template, the design change might be overwritten by the Designer Domino task each night. There may be other considerations as well. We keep it simple here to illustrate what needs to be changed.

4. To make the change take effect, the current view index needs to be discarded. You can wait for the nightly run of UPDALL to do this or you can do it manually by running the UPDALL command against the database from the Domino console². Refer to the Domino 5 Administrators Help Database for details.

Once that is complete, if you want to test the results, you can simply make some changes to the database and then close the database. You may have to wait several minutes for UPDATE to run. If you want to speed up the test, there is a notes.ini parameter you can use to control how UPDATE runs.

Normally, although UPDATE looks for database changes every 5 seconds, it does not act on them for about 5 minutes. This is called the *suppression time*. UPDATE does this to wait for more updates and to “batch them up”. On a busy server, this reduces the frequency in which UPDATE runs. You can change the default from 5 minutes to 1 minute by adding Update_Suppression_Time=1 to the notes.ini file.

Note: This should be a temporary change only.

² You may also use the Submit Domino Command (SBMDOMCMD) or Run Domino Command (RUNDOMCMD) commands.

7.4.7 Conclusions

The only way to predict what benefit you might achieve from making the changes described in this chapter is to actually measure the amount of time UPDATE is running and determine if there are any views that could be *reconfigured*. There have been some instances where CPU utilization has dropped as much as 45%. In others, the average CPU utilization was not affected much, but spikes in response time when UPDATE was rebuilding a view were greatly reduced.

7.5 TCP/IP between Domino servers on the same iSeries

In previous releases of OS/400 and Lotus Domino, if you had multiple Domino servers (that is partitions) running within the same physical system, we recommended that you use the loopback interface (LOOPBACK_TcpIPAddress). This gives you more efficient TCP/IP communications between multiple servers on a single physical system. Starting with OS/400 V4R4, the TCP stack now redirects all communications to the loopback interface if the destination address is on the local system. However, to allow this new function to work properly, you should now remove all references to the LOOPBACK interface in the Domino server by following these steps:

1. Remove the PORT definition in the Server document.
2. Remove any Connection documents that reference the Loopback port, or modify the documents to reference a valid port such as TCP/IP.
3. Remove any reference to the loopback port from notes.ini (this should have been removed automatically after performing step 1).

7.6 GTR search engine version 3.4

Domino R5.03 now includes an alternate *Global Text Retrieval (GTR)* search engine version 3.4. GTR is a set of functions which enables Domino to perform high performance full text search to a large set of documents. High performance full text search is realized by searching an index file which is created from source documents in advance.

7.7 Using the Extended Directory Catalog

The Extended Directory Catalog is a new type of directory catalog available in R5.0.5. For additional information on this feature and further details on setting it up, see the Release Notes, particularly the "Extended Directory Catalog" section in the "Things you need to know" section.

Some early deployment work at IBM Research with an IBM Worldwide Extended DirCat (EDC) indicates substantial (approximately 20%) CPU reduction that would also improve overall server performance (including the router).

7.7.1 Comparing EDC to a standard directory catalog (DirCat)

The improvement is relative to using a standard directory catalog, as the router can bog down in Full Text (FT) searches. The standard directory catalog is designed to be small, so it builds only one sorted view by last name or distinguished name at the Administrator's option, and uses FT search to find names that aren't matched in the view. For example, if the DirCat is sorted by last name, "Fred Bloggs/Accounts/MegaCorp" won't show up in the \$Users view,

but FT search will find it. On the client, the difference between delay when finding a value in a sorted view and when searching for it in a full-text index is trivial, partly because we can't send messages that fast! Multiplied by thousands of recipients on a server, the delay becomes a bigger problem. The Extended Directory Catalog (EDC), which is designed to be fast instead of small, addresses this by aggregating all the source directories into a single superdirectory with standard views.

The Router must still search the primary directory before searching any secondary directory, including an EDC. As with directory assistance, there's no necessity, and anyway you must *not* include the primary directory in the EDC.

7.7.2 Using Extended Directory Catalog to improve performance

Using an Extended Directory Catalog on a server is particularly recommended if using the standard Server Directory Catalog (created from the DIRCAT50.NTF template) has been determined to be causing Mail Router backups. When the Mail Router uses the standard Server Directory Catalog for mail routing lookups, it uses full-text (FT) searching, a slower process than using views, to look up names that don't correspond to the "Sort by" directory catalog configuration selection, as detailed above.

7.8 Investigating mail backlogs and performance issues

If mail routing or performance backlog issues develop, possibly as a result of using a standard or GTR Server Directory Catalog, try following these steps to troubleshoot the problem and implement the EDC:

1. To determine if the standard Server Directory Catalog is the cause of the problem, remove its file name from the Server document or from the Public directory profile.
2. If the standard Server Directory Catalog is the cause of the problem and the "Sort by" configuration selection is not Distinguished Name, change the selection to Distinguished Name and then re-enable the Directory Catalog to see if the change resolves the problem.
3. If using Distinguished Name as the "Sort by" selection does not solve the problem, use the Extended Directory Catalog instead, so that the Router uses view lookups rather than full-text searches to look up names.

Watch for agents running on mail files that do *not* have a full text index on the mail files. searching through each new mail item (for example moving junk mail to a waste bin). If the mail file is not full text indexed, a temporary full index will be built for each new mail items that arrives. See "View_Rebuild_Dir" on page 222.

7.9 Domino console logging

One concern with the Domino Console Log (buffer), when it comes to performance, is the actual size of the Domino console log. On the iSeries server, we recommend that the size of the Domino console log does not exceed 10,000 entries (the default). A Domino console log with more than 10,000 entries can result in poor response times for the Display Domino Console (DSPDOMCSL) and Work With Domino Console (WRKDOMCSL) commands.

The size is set in the iSeries environment variable Notes_AS400_CONSOLE_ENTRIES. The Add Environment Variable (ADDENVVAR) and Change Environment Variable (CHGENVVAR) iSeries server commands are used to maintain the size of the Domino console log. You have to perform the ADDENVVAR or CHGENVVAR command in the job that starts the Domino server before it is actually started (unless you use system-level environment variable).

For more information about iSeries environment variables, see 7.16, “iSeries environment variable settings” on page 224.

7.10 Number of users per Domino server

What is the maximum number of users supported on a Domino for iSeries server? This is a question we often heard since publishing a guideline³ related to Domino for AS/400 R4.6 in the first edition of this book. There is no straight answer except for saying: “No limitation, neither for registered users nor for concurrently active users exists.” Was the recommendation wrong? No, it was not.

With any version of Domino, there is a certain maximum workload that can be handled efficiently by each Domino server. With Domino R5, the implementation has been changed, so each server can support a much higher workload than it could with previous releases. Assume you observe the Domino response times while increasing workload on a powerful iSeries server. There is certain point, where hardware and operating system may still be able to support higher workload, but the response times begin to increase exceptionally because of contention within the Domino server.

This makes it desirable for many administrators to limit the workload and add partitioned servers, when there are still hardware resources available. However, the trade-offs in running multiple partitions are different on different platforms. On other platforms, there is a reluctance to run more than one, or at most two to four partitions on production servers for reliability and manageability reasons. On iSeries, because of the way Domino takes advantage of OS/400 architecture, there is no such limitation. Rather, it is viewed as an advantage. By dividing large numbers of Domino users into smaller Domino partitions, higher reliability is achieved and greater overall performance and capacity are realized. Now, how do you limit the Domino workload?

In a mail environment in a single organization, you may make two assumptions:

- ▶ The workload produced by each single user normally does not deviate extremely from the average.
- ▶ The percentage of concurrently active users may vary significantly for different times of the day, but not so much for a (randomly chosen) subset of the total number of users.

Keeping the above in mind, the workload generated is proportional to the number of registered mail users. Therefore, in a mail environment, for a relatively homogenous organization, you may limit the workload by simply limiting the number mail databases, in other words the number of *registered* users per server.

IBM Rochester, currently supports more than 10,000 Notes users on three iSeries servers and already started doing that in the early days based on a beta version of Domino 4.5. The recommendation for R4.6 mentioned above was a result of the experiences at IBM Rochester. After they migrated to Domino R5, they were able to raise the number of active mail users per Domino partition much higher.

Therefore, the Workload Estimator now recommends that you distribute the workload over multiple Domino partitions, if you have more than 3000 concurrently active users (see 3.4.2, “Number of partitions” on page 26). Keep in mind though, the optimum number may vary in both directions, depending on the working habits of your users and the complexity of your applications.

³ At the time, we recommended 700 registered or 500 concurrently active users.

7.11 Using a separate partition for the R5 SMTP MTA server

The R5 SMTP MTA process can be considered to be a pure batch task working on inbound and outbound SMTP mail streams. Outbound it takes Notes mail, converting it as appropriate to MIME, and transforming it into an SMTP outbound stream to send to the SMTP relay or firewall. On the inbound side, it receives SMTP and MIME data streams from totally unknown origins anywhere on the Internet, and then attempts to parse it and convert it into Notes mail. In most major organizations and enterprises, we recommend that the SMTP MTA is run in a separate partition (or on a separate server) to avoid it interfering with the Notes mail servers.

Processing SMTP MIME e-mail, large attachments, anti-virus checking, and anti-spam filtering can severely slow processing for “real” interactive users and jobs. Also, this type of “alien file and data” processing, is a high risk activity, much more prone to crashing and errors than the normal activity of a mail server because of the parsing and mixed file handling of SMTP data generated, basically, anywhere on the planet.

If you do have a need to keep the R5 SMTP MTA on a mail server, we recommend that you change the job run-time priority and time slice. See 6.5, “Choosing which processor priority to use” on page 173, and for detailed information, see Chapter 11, “Internet and intranet performance tips” on page 301.

7.12 Excessive translations on iSeries and zSeries

On the iSeries and zSeries servers, the whole operating system is based on EBCDIC, unlike the pSeries and xSeries servers, which are based on ASCII. Domino is based on Lotus Multi-Byte Character Set (LMBCS), a superset of ASCII, which works reasonably well translating with ASCII but sometimes is not quite as fast with EBCDIC.

7.12.1 Opening up any suspect job using the new Domino panel group

Make sure to sign on to the iSeries with a user profile having *ALLOBJ and *JOBCTL special authorities and follow the instructions given in “Display Call Stack: Panel group update to improve readability” on page 131.

This display shows whether you are spending a lot of time doing (excessive number of) “string translates” (StringXlate).

Figure 7-12 shows a sample call stack with the string translate clearly shown in the sequence of procedures:

```
“LCStreamConvert StreamTranslate StringXlate“
```

Display Call Stack			
		System:	AS06
Job: SERVER	User: QNOTES	Number:	017486
Thread: 00000005			
Procedure	Statement	ILE Module	ILE program
LE_Create_Thread2__FP12crtth_parm_t	0000000826	QLECRTTH	QLESPI
pthread_create_part2	0000001004	QPOWPTH	QPOWPTH
ThreadWrapper	0000000008	THREAD	LIBNOTES
Scheduler	0000000032	SCHED	SERVER
DbServer	0000000224	DBSERVER	SERVER
ServerNoteOpen	0000000029	SVNTOPEN	SERVER
NSFNoteOpenExtended	0000000044	NSFSEM3	LIBNOTES
EMAfter	0000000011	EXTMGR	LIBNOTES
EMDispatch	0000000025	EXTMGR	LIBNOTES
NoteOpenHandler	0000000050	LNPEXT	LIBDECSEXT
FetchDocument	0000000006	LNPEXT	LIBDECSEXT
ExecProcedure	0000000026	LNPEXT	LIBDECSEXT
LCConnectionCall	0000000001	STATE	LIBLCAPI
retry_LCConnectionCall	0000000039	STATE	LIBLCAPI
LCXCall	0000000059	DB2	DB2
GetMetadata	0000000028	DB2	DB2
StoreData	0000000135	DB2	DB2
LCStreamConvert	0000000018	STREAM	LIBLCAPI
StreamTranslate	0000000058	STREAM	LIBLCAPI
StringXlate	0000000002	STREAM	LIBLCAPI
LCProcCritSecWait	0000000003	MLTPROC	LIBLCAPI
pthread_mutex_lock	0000000900	QPOWPMUT	QPOWPTH
lockSlowWithInitAllowSig__9Qp0wMutexFv	0000000104	QPOWMTX	QPOWMTX
Bottom			
F3=Exit	F10=Update stack	F11=Display instruction	F12=Cancel
F16=Job menu	F17=Top	F18=Bottom	F22=Display entire name

Figure 7-12 Sample call stack viewed with the new Domino panel group

Note also “pthread_mutex_lock” and “lockSlowWithInitAllowSig__9Qp0wMutexFv”, which are where the delays and performance problems show up. If you determine from the call stack that Domino is spending a lot of time doing a high number of string translates, isolate the active databases (.NSFs) and template files (.NTFs) involved and display the attributes of the databases.

Figure 7-13 shows the alog.ntf attributes.

Display Attributes	
Object	/domino/applsrvr/alog4.ntf
Type	STMF
Owner	QNOTES
System object is on	Local
Auxiliary storage pool	1
Object overflowed	No
Coded character set ID	819
Hidden file	No
PC system file	No
Read only	No
Need to archive (PC)	Yes
Need to archive (AS/400)	Yes
More...	
Press Enter to continue.	
F3=Exit F12=Cancel F22=Display entire field	

Figure 7-13 Display Attributes of a Notes file alog4.ntf showing a CCSID of 819

If the database does not have the same Coded Character Set ID (CCSID) as your local iSeries, that is your local country or region's character set (in the USA that would be 37), it may need switching. In the above example, the CCSID is set to 819 (that is the ASCII CCSID). This may be the cause of some of your high number of translates. Unnecessary translation efforts will cause performance bottlenecks.

Shut down Domino, use the WRKLNK option 3 (Copy) on the database (.nsf or .ntf) in question, and alter the CCSID.

Figure 7-14 shows the prompt of the Copy (CPY) command preparing to change the CCSID of the file to 37.

Copy Object (CPY)

Type choices, press Enter.

Object mynotes.nsf

To directory '.'

To object mynotes.nsf

Symbolic link *NO *NO, *YES

From CCSID 819 1-65533, *OBJ, *PCASCII...

To CCSID 37 1-65533, *OBJ, *CALC...

Data Format *BINARY *BINARY, *TEXT

Directory subtree *NODIR *NODIR, *NONE, *ALL

Replace object *YES *NO, *YES

Owner *KEEP *NEW, *KEEP

Bottom

F3=Exit F4=Prompt F5=Refresh F12=Cancel F13=How to use this display

F24=More keys

Figure 7-14 Copy Object command preparing to switch from a CCSID of 819 to 37

Note: Either use parameter OWNER(*KEEP) or, after copying the file, reset the Owner back to QNOTES.

Once you have checked to see that all your databases are now correctly set to the right CCSID, if you still have high string translate counts, start looking at the application coding itself, isolate which agents are causing the issue, and then focus in on the actual code. Is there a high use of simple STRINGS in the code rather than numeric, date, and binary fields as appropriate? Perhaps it is now time to review your coding standards and modify the code based on best industry practice rather than convenience.

7.13 Optimizing mail: White space, compact, and quotas

This section looks at the two main compact styles, whether to recover white space, use quotas, and why you should have a mail archival policy.

Using COMPACT -b to maintain white space and reach steady state

Here are the latest recommendations on the issue of recovering the white space from Notes databases, particularly mail files to stop the server running out of space each week.

Do not recover space with COMPACT

We do not believe that our customers are actually doing the correct thing, in running COMPACT with white space deletion (COMPACT -B or -C). With TxL enabled, our very strong recommendation is to *not* remove white space for various reasons, not the least being the fact that the *Database Instance ID* (DBIID) will be changed forcing a full backup to be run immediately, and that the database must be shut down and offline to compact copy style. (For

more information on the DBIID, refer to 13.5.1, “What is the Database Instance ID (DBIID)” on page 385.) We believe that even on a busy mail server, it is not necessary to COMPACT -B (or -C) an active mail database to reduce the file size (with or without TxL enabled!) as long as an archiving policy is also in force.

COMPACT and white space

Note: The parameters for the COMPACT command are case sensitive. That is, COMPACT -B and COMPACT -b perform differently.

When COMPACT -b runs on a mail file, or indeed any database, rather than decrease the physical size of the database, we now recommend that you defragment the database from within, re-organizing all the records down to one end of the disk space, and shuffling the empty spaces (white space) up to the other end. COMPACT -b reorders data in a database so that the data is consecutive. The percent utilization goes up while space is compacted within the NSF structure. The file size remains the same.

By defragmenting the database, the white spaces become a single large “white space”, and therefore, we now have room for larger incoming items, mail, attachments, etc. With this new incoming “data”, we can now accommodate within the existing database without having to “expand” the database by adding additional new blocks to the end.

The database will gradually again become more fragmented through use. As random records (notes) are deleted, creating smaller white space holes, the database becomes increasingly fragmented. Consequently larger incoming notes and files, attachments, etc., will no longer fit within any spaces inside the database, and so the database must expand itself.

Important: Frequent inline COMPACT -b defragmentation is required to keep the white space in a usable, contiguous form for as much of the time as possible.

Re-using white space

For pure performance reasons, Notes always re-uses white space within a database before allocating additional disk blocks of storage. The rate of perceived growth of any database will slow down, and eventually stop, as the amount of white space created by COMPACT -b approaches the actual storage required for new messages (or notes etc.) delivered during the day or days between compactions. This is called *steady state*.

COMPACT -b nightly reduces fragmentation

If you ran COMPACT -b nightly for a week or two, you would be pleasantly surprised that, rather than seeing the disk reaching that “full” point on the Friday before your usual COMPACT -B, it would peak and then achieve “steady state”, probably below your previously noted high water marks.

Massive fragmentation over a week can cause apparent size problems on your system. We recommend that you avoid this. The continual gentle online defragmentation achieves steady state below or close to your previous watermarks.

COMPACT -b nightly can run online, speed up access, allow more users

COMPACT -b can also be run online without affecting live mail or other database users (it might go slower for a minute or two, but no shutdown, etc. is required). Obviously you can choose to run it overnight or at quiet times. You may even want to run it more than once per day. Remember, the more efficient the database structure is, the faster the system can find and serve up the data, and the lower the overall system loading will be.

Your users will also be pleasantly surprised that mail access feels “quicker” every day of the week, instead of just on Monday morning. That’s because the data has been re-sequenced within the database, and can now be accessed more quickly, so maybe you can even add a few more users!

Why make work for yourself copy compacting your databases?

If you really want, you could always run a COMPACT -B every month or two. But if you do, you must then run a set of full backups because all the DBIIDs and you are using Archival TxL for your backups. But by regularly running COMPACT -b, you have achieved steady state long before then, and the nightly COMPACT will keep things in order.

Steady state nirvana

Let’s say you follow the above recommendation, after you reach steady state, and then did an operating system level backup, delete, and restore of all databases and files (that is the old fashioned backup-restore defragmentation of the disk). In this case, each database would now have all its data *and* all its white space in sequentially allocated blocks directly associated with it, not scattered. Therefore, performance would be markedly improved in the I/O area. We suggest doing this after you reach steady state. We recommend doing the backup-restore defragmentation *after* a full backup and then use disk-to-disk copy for the backup/restore (with Domino down).

Even without this drastic step of backup, delete, and restore, leaving white space alone to reach steady state would at least halt the march toward total fragmentation. This would occur by the totally unnecessary freeing up and then immediately re-allocating of new blocks by “freeing up” white space to the operating system.

Table 7-1 shows COMPACT command line options for compacting with and without white space recovery.

Table 7-1 COMPACT command line options for COMPACT

Option	Command-line equivalent	Description	Domino Release
Compaction style: In-place (recommended)	-b	Uses in-place compaction and recovers unused space without reducing the file size, unless there’s a pending structural change to a database, in which case copy-style compaction occurs. This is the recommended method of compacting.	New to R5
Compaction style: In-place with file size reduction	-B	Uses in-place compaction, recovers unused space, and reduces file size, unless there’s a pending structural change in which case copy-style compacting occurs. If you use transaction logging, do full database backups after compacting completes.	New to R5
Compaction style: Copy-style	-C Upper or lower case “C” will work.	Uses copy-style compaction. Use this option, for example, to solve database corruption problems with Release 5 format databases.	New to R5

Implementing mail archiving

Once an archival policy is implemented to restrict indefinite growth of users' mailboxes, all users will settle into a pre-defined mail file size. Eventually all databases will reach steady state, freeing and re-using their own white-space, and taking the laborious task of managing and worrying about data growth away from the administrator.

Mailbox quotas

Setting mailbox quotas to control disk usage is like using a car crusher to keep a car park tidy. It makes little or no sense. In attempting to save fractional amounts of cash on disk space because you did not organize an archival policy, your company suffers for the following reasons:

- ▶ You indiscriminately prevent your users from being able to do their productive job at your corporation, by stopping them from sending outbound mail, and possibly receiving inbound mail, until they administrate their own mailbox in peak busy corporate time.
- ▶ Your e-mail rejects inform all your external suppliers, customers, and e-mail callers to these people that this person's e-mail has been shutdown, causing outsiders to think the user has moved to another company and they should take their business elsewhere.
- ▶ You are advertising on the Web that you have no control over your users or your IT budget or infrastructure by sending out unnecessary e-mail rejection notes almost at random.
- ▶ You must employ additional IT staff to cope with the complaints, run additional backups, and restore mail hastily deleted to send out an urgent e-mail and manually "copy compact" the user's inboxes *after* they have deleted mail that they may actually need in future. Where TxL is enabled, you then have to schedule a full backup for that mailbox immediately if recovery in the future is to be possible.
- ▶ Potentially each user could spend two hours per week tidying their mail file, on an average wage of \$30,000 per year (\$15/hour), in a company with 1000 employees that is a cost of \$30,000 per week, or \$1.5m per year. Does that disk cost savings still look cheap to you? Would you like to estimate the cost of actual lost business during that time? In the above example, in person hours alone, it comes to 11,250 hours spent tidying the mailbox (that is work that IT should own), which is the equivalent of seven employees. Are these figures still looking good to you?

Quotas can cause restore to totally fail

If the option *Obey database quotas during message delivery* is enabled in the configuration document in the Domino Directory, the database can be recovered from backup to the size of the database quota.

Once the quota is reached, however, any replayed transactions from the transactional log extents cannot be applied to the database past the limit of the quota. This results in lost transactions and lost data. Furthermore, the database quota cannot be increased until the database is fully restored. Since transactions cannot be applied over the database quota, a full database restoration is never achieved.

Quotas add to the overhead and complexity of restores

If *Obey database quotas during message delivery* is disabled, a database quota has been exceeded and the database must be recovered for any reason. During recovery, the transaction or transactions would be replayed into the database (for example, router delivering mail), so the database would be restored to its original state (which is over quota).

To work around the issue, the database must be brought under quota and then compacted to reduce the size of the database. When the problem occurs, however, the default compaction style changes from copy style to in-place compaction (with no file reduction) for any databases that are transactionally logged.

If users delete documents to bring the database under quota and compact it, the database will still remain over quota since in-place compaction is being used and the file size is not being reduced. The administrator must then compact the database using copy style compaction. When this is done, however, it changes the DBIID. As a result, a full backup must then be taken.

Mail archival policy

Use the standard R5 archival utility. Run automated archiving on all mail over 30, 60, or 90 days. Place all archived mail on an archive server (never the mail server). Add an icon to the user's desktop, add an archive to the mailbox, and allow users to replicate their archive to local or access it on an archive server. Show users where "old mail" goes, since that is all they need to know. You only need a very modest system for an archive server because not many users will ever access it at any one time. It may, however, need quite a bit of disk space! If you really want, you can also archive the archives to CD for all mail that is over 12, 24, or 36 months old and give user a copy of the CD (keeping a copy in your off-site warehouse).

Legal advantage of an archival policy

Together with TxL keeping all the mail, a simple archival policy helps you when you are legally required to recover selected mail items from months or years ago by your legal department or other legal agency. Because you have a sensible mail backup and archival policy, this is very easy to do and everyone is happy with your IT department. If you use the quotas, the only one you will eventually find that you fulfill is part of the quota of staff to be made redundant in your department.

For more information, see the following documents:

- ▶ *Why you need transaction logging* white paper in the Technical Library:
<http://www.lotus.com/home.nsf/welcome/itcentral>
- ▶ Technotes: For more information, refer to <http://www.support.lotus.com> and search for the following Technote numbers:
 - Domino R5 Compact Not Reducing the Size of the Database #173311
 - Transaction Logging in R5 #172508
 - How Do You Check Your Transaction Log Statistics? #173421
 - Commonly Asked Questions About Transactional Logging #177785
 - How to Utilize Transactional Logging Statistics #177796
 - "Circular" Versus "Archive" Transactional Logging in Domino 5.x #179363
 - Transactional Logging and How it Operates #179858

7.13.1 Mail file size

"So what's wrong with my 800 MB mailbox? I might need all my old e-mails one day."

Let's quickly go back to the basics and review how Domino databases react to change. Each addition, deletion, or change to a Notes record forces the active views to be rebuilt to reflect the new state of that index. With a mail file, every time a new mail item arrives, the view is rebuilt. The more items there are in the view, the more mailboxes there are on the system, and the more I/O and CPU are required to rebuild all those separate views. Imagine what an "All Staff" 1 MB memo does to the mail server. Multiply the amount of mail you expect to receive daily, by the average number of old e-mail in your inbox, by the number of mailboxes

on your system. That is the work factor you are imposing on Domino. Now reduce the mailbox size to a reasonable number and re-calculate. From those savings, you can either add another 20% more users or enjoy far better server performance. Very simple. Small well compacted mailboxes perform several hundred percent better than large ones do.

7.14 Domino memory management

Domino is an application suite that has its own work and memory management functions that are a layer on top of the iSeries work and memory management. Figure 7-15 shows how Domino memory management is performed. A certain amount of main storage is set aside for:

- ▶ NSF buffer (=indexes)
- ▶ Database Cache (=document data)
- ▶ Application code
- ▶ Users

	NSF Buffer	DB Cache	Appli- cations	Users

Figure 7-15 How Domino divides the main storage

All of these memory areas are dynamically adjusted by Domino. They both grow and shrink during a day.

There are a number of variables that we recommend setting in the notes.ini file. Unless you explicitly add these variables in your notes.ini file, the default values for each of these variables are used, and in some cases, this is not what we want. Your Domino server needs to be restarted after these entries are added to the notes.ini file to use the changes.

Use the **show stat** command on the Domino console to find out the resource utilization. Change the notes.ini setting accordingly.

Hint: Use a more specific **show stat** command instead of showing all the statistics available every time. For example, **show stat database** produces the following output.

```
COMMAND SENT: sh stat database
Database.BufferControlPool.Peak = 130,812
Database.BufferControlPool.Used = 48,740
Database.BufferPool.Maximum = 502000000
Database.BufferPool.Peak = 4780800
Database.BufferPool.PerCentReadsInBuffer = 98
Database.BufferPool.Reads = 702753
Database.BufferPool.Used = 4091136
Database.BufferPool.Writes = 30648
Database.DbCache.CurrentEntries = 0
Database.DbCache.HighWaterMark = 0
Database.DbCache.Hits = 0
Database.DbCache.InitialDbOpens = 121278
Database.DbCache.Lookups = 0
Database.DbCache.MaxEntries = 723
Database.DbCache.OvercrowdingRejections = 0
```

```
Database.NIFPool.Peak = 196,218
Database.NIFPool.Used = 128,438
Database.NSFPool.Peak = 130,812
Database.NSFPool.Used = 42,546
```

The following sections offer a few descriptions of the key notes.ini variables. They include a table of all the performance related variables and our recommendations on values that may need to be changed and suggestions on how to calculate a suitable and safe value for each.

7.14.1 NSF_Buffer_Pool_Size or Nsf_Buffer_Pool_Size_mb

The NSF_Buffer_Pool_Size and NSF_Buffer_Pool_Size_MB parameters let you specify the maximum size of the NSF buffer pool. This pool is a section of memory dedicated to buffering I/O transfers between the Notes Index Facility (NIF) indexing functions and disk storage.

We recommend that you set both NSF_Buffer_Pool_Size and PercentAvailSysResources (see “PercentAvailSysResources” on page 213) for all Domino R5 iSeries servers. This includes all non-partitioned and partitioned servers, because the Domino servers run out of their own storage pool, which is typically changed dynamically by the auto-performance adjuster (WRKSYSVAL QPFRADJ).

Previously if you did not set NSF_Buffer_Pool_Size, Domino would calculate a value to use based on the total physical memory on your system (taking roughly 3/8). To prevent Domino from accidentally stealing huge amounts of memory on very large iSeries servers (when none of the NSF_Buffer_Pool_Size parameters were set), the iSeries for Domino sets the notes.ini value to 300 MB during installation if it was not previously set.

Since the iSeries uses single level (memory) store, the system does not really dedicate the NSF_Buffer_Pool_Size to the server. In reality, it just forces higher page faulting when using it. We have seen systems with multiple servers where the total of all NSF_Buffer_Pool_Size was well above the machine's main storage, and it still performed amazingly well. Performance was degraded somewhat (about 10%) due to the rather high faulting rate, but nothing failed and the customer seemed very happy with it.

However, we much prefer to be more scientific and logical about memory allocation and usage, and control memory using these two parameters. There are a number of memory buffers used by Domino that are allocated based on one or the other of these parameters. Therefore, it is best to specify both to carefully worked out values.

This default value of 300 MB may work fine for most servers, but a better value can and should be calculated. It is even more important to set this variable if your Domino applications use a lot of indexing. The recommended setting for R5 varies depending on the number of partitioned Domino servers that are being used.

Here are easy steps to follow to calculate the NSF_Buffer_Pool_Size_MB on iSeries servers:

1. Find out from which storage pool the server is running. You can find this by using the WRKACTJOB command. The default is pool 2.
2. Find the size of the pool while your system is running in steady state (that is when QPFRADJ isn't changing the pool sizes). You can see this by running the WRKSYSSTS command.
3. If you multiple partitioned servers are using that pool, divide the pool size by the number of servers using the pool (excluding the *HTTPSETUP server) and multiply by 3/8. Set NSF_BUFFER_POOL_SIZE_MB in notes.ini to this number.

Figure 7-16 shows the four partition Domino on iSeries server.

```

Opt  Subsystem/Job  Type  Pool
      DOMIN001      SBS    2
      QNNINSTS      BCH    2
      REPLICa      BCI    2
      ROUTER        BCI    2
      SERVER        BCI    2
      DOMIN002      SBS    2
      QNNINSTS      BCH    2
      REPLICa      BCI    2
      ROUTER        BCI    2
      SERVER        BCI    2
      DOMIN003      SBS    2
      HTTP          BCI    2
      QNNINSTS      BCH    2
      SERVER        BCI    2
      DOMIN004      SBS    2
      HTTP          BCI    2
      QNNINSTS      BCH    2
      SERVER        BCI    2

WRKSYSSTS at steady state shows pool 2 is 3207 M
System  Pool  Reserved  Max  -----DB-----  ---Non-DB---
Pool    Size (M)  Size (M)  Active  Fault  Pages  Fault  Pages
1       340.52   167.89   +++++  .0     .0     .0     .0
2       3207.63   .17      400    .0     .0     4.5    6.3
3       35.83    .00      1      .0     .0     .0     .0

```

Figure 7-16 WRKACTJOB shows that all four Domino servers running out of pool 2

Since there are four servers, NSF_BUFFER_POOL_SIZE_MB should be:

$$3208 / 4 * 3/8 = 300 \text{ MB}$$

There are four things to check when fine tuning the NSF_Buffer_Pool_Size parameter:

1. Monitor how much NSF buffer pool your server uses by running the SHOW STAT DATABASE command. If BufferPoolPeak is routinely more than 95% of the maximum, then the NSF_Buffer_Pool_Size may need to be increased. If you have sufficient main memory, some memory may be freed from some other over-allocated task or job. Otherwise, you may have to consider buying additional memory or upgrading the system.

Here are the statistics you will want to obtain:

- Buffer pool maximum – (Database.BufferPool.Maximum)
 - Buffer pool used – (Database.BufferPool.Used)
 - Buffer pool peak – (Database.BufferPool.Peak)
2. Monitor Database.BufferPool.PerCentReadsInBuffer. When tested under a heavy workload, BufferPoolPeak was seen at 100%, while PerCentReadsInBuffer was under 95%. Under a light load, BufferPoolPeak was under 10%, while PerCentReadsInBuffer was over 97%.
 3. Monitor non-database page faulting by using the WRKSYSSTS command or a performance monitor. High faulting rates indicate memory contention, and you may be able to actually improve performance by decreasing NSF_Buffer_Pool_Size.
 4. Monitor the Mail Router's database cache size (which defaults to NSF_Buffer_Pool_Size_MB * 3). You should monitor Mail.DBCacheEntries to see if your

system is reaching the maximum available cache size. Then you can compare Mail.DBCacheHits to Mail.DBCacheReads to see how effectively your cache is being used

Note: The only difference between NSF_Buffer_Pool_Size and NSF_Buffer_Pool_Size_MB is that NSF_Buffer_Pool_Size is set in bytes. It also has a maximum possible value NSF_Buffer_Pool_Size_MB that is set in MB.

For example, the following two parameters are equivalent:

```
NSF_Buffer_Pool_Size=104857600 (100x1024x1024)
NSF_Buffer_Pool_Size_MB=100
```

Tip: You can create shared pools on the iSeries server and assign a Domino server instance to run in that pool. This allows performance adjustment to increase the shared pools storage when and if the base pool is not using the memory allocated. This way, a Domino server could be granted access to additional memory, when required when, other processes on the system do not require it.

However, even when using performance adjustments, you must be certain that you have defined the pool to allow the server to function properly. Undersizing a shared pool can result in poor performance as the server becomes hindered from a lack of memory.

PercentAvailSysResources

In Domino Release 5.0.4 or later, you can control the amount of memory allocated to each Domino server (subsystem) by percentage using the PercentAvailSysResources variable.

Before the introduction of this variable in 5.0.4, each Domino server assumed it had 100 percent of system physical memory available to it. And, unless you set an actual value in NSF_Buffer_Pool_Size, Domino would take three eighths (3/8) of total physical memory. Other Domino memory allocation sizing parameters would take a percentage of the remaining five eighths (5/8) of all memory. On iSeries, this behavior was prevented by defaulting the notes.ini NSF_Buffer_Pool_Size parameter to 300 MB, where it was not set during installation.

With the advent of the PercentAvailSysResources variable with Domino 5.0.4, you can now assign a portion of total system physical memory to each Domino server by specifying a value from 2% to 100%, which represents an absolute percentage of the system's total physical memory. On the iSeries server, memory calculations are based on the main storage size multiplied by PercentAvailSysResources if the Domino subsystem is running in *BASE pool or on the actual pool if running in a shared (for example *SHPOOL01) or private pool.

NSFBuffer Pool defaults to approximately three eighths (3/8) of that calculated base for each server. Note also that if QPFRADJ is not set to zero, and you are using shared pools, your shared pool sizes will be adjusted on the fly. That may affect how PercentAvailSysResources works in cases where the Domino server is started or restarted when the subsystem is already active for a longer time and the pool size has been adjusted meanwhile. In that case, using NSF_Buffer_Pool_Size will be more predictable.

The NSF_Buffer_Pool_Size (if omitted) is the time allocated three eighths (3/8) of the total amount of memory allocated to Domino by PercentAvailSysResources. If you need more memory in the NSF_Buffer_Pool_Size, you must adjust any one (or more) of the following areas:

- ▶ Physical memory (so that the amount from the same percentage increases)
- ▶ Proportion of memory that this partition acquires by increasing PercentAvailSysResources

- Deliberately set `NSF_Buffer_Pool_Size` to a higher value than three eighths (3/8) of available memory

In the first two cases, the remaining memory buffer allocation parameters (the 3/8 versus 5/8) will keep their correct relative values and balance. With the third case, you should carefully make certain that the `NSF_Buffer_Pool_Size` fits in with your actual requirements for this particular configuration by calculation if you set it to greater than 3/8 deliberately. Setting both variables will protect the overall system from Domino's stealing habits, if, at some future point more physical memory is added with no (Domino `NOTES.INI`) parameter adjustments.

You can find details on all the principal system performance parameters in Table C-3 on page 444.

For example, if you want to dedicate 25 percent of total system memory to a Domino server (in its own dedicated partition, typically), add the following line to the `notes.ini` file:

```
PercentAvailSysResources=25
```

This effectively leaves 75 percent of memory for other Domino partitions or applications.

For example, a Domino server on a system with 10 Gb of memory and a Domino server with its `notes.ini` setting `PercentAvailSysResources=25` would reserve 2.5 Gb for itself.

The `PercentAvailSysResources` parameter has significant advantages over the old method of “hard coding” the memory using `NSF_Buffer_Pool_Size` parameters:

- If you add additional memory, it is immediately proportionally shared between the Domino servers without any need to change or edit the `notes.ini` parameters.
- In high availability clustering and failover scenarios, when a Domino server's workload is failed-over to another partition, the percent of physical memory required by the receiving partition to cope with the increased workload is quite simply the sum of the two separate `PercentAvailSysResources` parameters from the failed and the newly joined partition.

Example with four Domino partitions sharing with other applications

For example, if you have a Domino server with four partitions and want to control the memory resources allocated to each partition, you could set the `notes.ini` of partition 1, 2, and 3 with the line `PercentAvailSysResources=20` and the `notes.ini` of partition 4 with the line `PercentAvailSysResources=40`. Here you should make sure that the sum of the values you specify in each partition do not exceed 100 percent! If it does, the systems will simply steal from each other, which could cause unnecessary thrashing and paging.

To reserve system memory for other applications, you can choose values that total less than 100 percent.

Indexing and costs associated

If your Domino applications use indexing, it is tempting to think that you can simply increase the size of the NSF buffer pool and dramatically enhance server performance. However, just as adding physical memory to a system does not always boost performance, neither does designating a bigger NSF buffer pool necessarily produce dramatic results. (See the article “Putting the right spin on Domino Performance” Parts 1 and 2, which is available from the Iris Today Web site at: <http://www.notes.net/today.nsf>).

Using a buffer pool summary

Domino does a good job by itself of allocating memory to the NSF buffer pool by calculating a default setting based on fixed overhead and total memory available. (Depending on the Domino version and other factors, the default usually amounts to between 25 percent and 33 percent, that is, the now famous 3/8 of system memory). What's more, changes to the NSF buffer pool allocation algorithm in R5 mean that the Domino server uses only the amount of memory it needs, up to the notes.ini NSF_Buffer_Pool_Size (default 300 MB on iSeries), and uses the amount allowed by PercentAvailSysResources. The R4 strategy pre-allocated and reserved the entire default or specified amount of memory. But in R5, when you specify a value of 600 MB for the buffer pool size and only 200 MB is needed, only 200 MB remains allocated. In Domino R4, if you specified a value of 600 MB, then all 600 MB was immediately treated as all allocated by Domino.

Notes:

- ▶ The NSF buffer pool setting in R5, up to and including 5.0.3, should not exceed 1920 MB (or 2,013,265,920 bytes).
- ▶ If you have Domino Release 5.0.4 or later, you should use the PercentAvailSysResources variable to allocate memory on the iSeries.

To summarize, we recommend that you implement the PercentAvailSysResources notes.ini variable to control total and overall memory allocation. For releases prior to 5.0.4, where PercentAvailSysResources is not available, there are some scenarios where setting non-default values for the NSF_Buffer_Pool_Size variable can prove useful. After 5.0.4, we still recommend that you set NSF_Buffer_Pool_Size together with PercentAvailSysResources.

If you employ any of these notes.ini variables, test your server performance to see if you get the results you expect. The number of users, size and number of views, and number of databases all factor in when determining what the settings should be. And remember to review the settings when the system configuration changes or after upgrading the system to a new Domino release, because it is likely that you will need to adjust them.

You should always test and monitor with a realistic data set before you make any configuration changes to your production environment. To help you do so, we recommend the following actions:

- ▶ If you are running 5.03 or higher, enable platform statistics by adding platform_statistics_enabled=1 to the notes.ini file.
- ▶ Enable STATREP collection on the server.
- ▶ While the server is under load, issue the **sh stat** command in 15 minute intervals. This dumps information about memory usage to the console.
- ▶ Review the data.

Table 7-2 lists the SH STAT database.* statistics and their utilization.

Table 7-2 SH STAT output parameters and a description of their content and use

Parameter	Description
Database.BufferControlPool.Peak	Peak number of database control pools used.
Database.BufferControlPool.Size	Size of database buffer control pool.
Database.BufferControlPool.Used	Number of database control pools used.
Database.BufferPool.Allocated	Number of database control pools allocated.
Database.BufferPool.Maximum	Maximum size of database control pools.
Database.BufferPool.Peak	Peak number of database buffer pools.
Database.BufferPool. PerCentReadsInBuffer	Percentage of read requests satisfied from the buffer. In performance terms, this is “cache hits”. Note: This value will be low for quite a time after a server starts as the cache is being populated. This value should not be referred to if the server has not been running for at least two hours.
Database.BufferPool.Reads	Number of database buffer pool reads.
Database.BufferPool.Used	Number of used database buffer pool.
Database.BufferPool.Writes	Number of database buffer pool writes.
DataBase.DbCache.CurrentEntries	Number of entries in the database cache at this time.
DataBase.DbCache.HighWaterMark	High water mark of the database cache.
DataBase.DbCache.Hits	Number of hits to the database cache.
DataBase.DbCache.InitialDbOpens	Number of database opens done by the database cache.
DataBase.DbCache.Lookups	Number of lookups to the database cache.
DataBase.DbCache.MaxEntries	Maximum number of entries in the database cache
DataBase.DbCache. OvercrowdingRejections	Number of rejections due to the overcrowding of the database.
Database.ExtMgrPool.Peak	Peak number of external manager pools.
Database.ExtMgrPool.Used	Number of External Manager pools.

Notes: Where statistics suggest that the NSF buffer pool size should be increased, the amount of the increase should not exceed 10%. Be sure to review the data again.

Refer to the StatRep log to see if brief pairs or patterns of spikes are a “normal” daily trend at particular times and can therefore be safely ignored in the short term.

Server_Max_Concurrent_Trans

This parameter regulates the number of concurrent transactions that can occur in a server at one time. When the number of concurrent transactions is reached, the other transactions are put into a wait state until one of the active transactions completes. In R4, we recommended that you set this parameter to -1, but in R5 on iSeries, you must *not* use -1.

In R5 Domino now fully utilizes thread pools. These pools are set by default to forty threads per port. However, if you set `Server_Max_Concurrent_Trans` in `notes.ini` (default is 20), be aware that IOCP threads default to two times this number (that is 40 if the default is set). If `Server_Max_Concurrent_Trans` is set to -1 (representing the maximum), Domino will create twice that number of IOCP threads, that is, thousands of threads for the server pool. You may then see:

- ▶ Server job taking up a lot of CPU
- ▶ Thousands of threads running under the Server job
- ▶ High faulting and paging in Domino memory pool
- ▶ Slow performance
- ▶ Users unable to connect

Important: Be absolutely certain this parameter is not set to -1. We recommend that you do *not* set this parameter unless specifically advised by IBM Support.

Server_MaxSessions

This parameter defines the number of user sessions a Domino server can accommodate. When a new user attempts to log on, if the current number of sessions is greater than the value of `Server_MaxSessions`, Domino closes the least recently used session. For a session to be considered for closing, it must have been inactive for at least one minute.

Important note: Reducing `Server_MaxSessions` to a low number will never prevent Domino from allowing more than that number of concurrent active users on the server. Reducing `Server_MaxSessions` will, however, cause sessions to start dropping very soon after they become inactive. This technically frees up resources, but becomes very heavy on the CPU and system in re-creating the sessions for all the same users slightly later. Conversely, Domino will not close any session that has been idle for less than a one minute regardless of the demand on the server.

There is presently no way to prevent a user from opening a new session on a Domino server (for more information, see 10.2, “Clustering” on page 278). If there are too many concurrent sessions (active sessions) on the Domino server, errors may occur.

Reaching this limit on iSeries can cause the `SERVER` task to start taking up a lot of CPU resources with a single thread, usually the one listening for connect requests on the TCPIP port, trying to get a lock on a semaphore. Users see the message “Remote system no longer responding”.

Server_MaxUsers

This parameter specifies the maximum number of active NRPC user sessions allowed on a server. This is *only* effective for NRPC (Notes Client) users and does not affect HTTP users or HTTP sessions. The HTTP user is accessing the server via an ICM server allowing HTTP users to continue to create new sessions even after this limit is reached.

When this number is reached, the server state becomes “MaxUsers” and the server stops accepting new Database Open requests from NRPC users. This prevents new sessions from starting and stops existing NRPC users from opening another database. They also see the error message:

Server Error: Access to the server is restricted due to maximum number of users.

Servers that are not in a cluster reject the access requests.

When using the ICM for failover and workload balancing, here as in standard Domino clusters. Domino computes the server availability index based on all open sessions, whether they are from Notes clients, HTTP clients, or other Domino servers. To configure workload balancing and failover, use the same settings, that is `Server_Restricted`, `Server_MaxUsers`, and `Server_Availability_Threshold`. For stopping database availability, you would mark a database out of service or pending delete. We recommend that you set the default to 0 (unlimited access to server by user).

Server_Session_Timeout

The `Server_Session_Timeout` parameter from the `notes.ini` file causes Domino to terminate an NRPC network or mobile connection/session that has been idle for this amount of time. A session is considered idle if no activity detected. If it is set too low, the server may have to re-open database server sessions too often. The best setting for this parameter is determined by the server load, and numbers of concurrent users on the server, etc.

The session timeout only applies to Notes clients and not to Web browsers. The Domino HTTP task accesses the server directly or via an ICM server so HTTP users are not affected by changes to this time-out value.

R5 has much improved handling on concurrent user sessions over R4, so this parameter can be safely removed. It has no affect unless the server is busy. Be aware, creating a new session is very CPU expensive. Re-using sessions is *much* more efficient, which is why we have the parameter.

The term “Session” also includes server “internal” sessions, such as agents, replicator tasks, server to server, and so on, so plan for it!

Consider this example. If you have 60 agents running every 35 minutes, and you set this timeout to 30 minutes, you will unnecessarily create one session per minute because every agent will be timed out!

The default for this parameter is 240 minutes (4 hours). We recommend that you reduce it to 30 to 45 minutes, but not less.

iSeries parameter TCPKEEPALV

The iSeries has a TCP parameter that specifies the time (minutes) that TCP waits before sending out an idle tickler probe to the other side of a connection to test for signs of activity. On the iSeries, this is the `TCPKEEPALV` parameter.

The default value for this parameter is 120 minutes. Setting this value to a smaller number will cause sessions to be cleaned up sooner. If the tickler finds the “other half” of the conversation has “gone away”, that is, it does not respond at all, it is cleaned up. It should be noted that a TCP/IP conversation that responds at the tickler level, but not at application level will never be closed by TCP/IP, until either the application closes the conversation or the conversation is ended abnormally by OS/400.

Use `CFGTCP` option 3.

We recommend that you set this parameter to 15 minutes.

Translog_MaxSize

This parameter defines the maximum size in MB that the collection of 64 MB Circular TxL log files can grow to before being forced to start overwriting (re-using) themselves. This parameter and `Translog_UseAll` are both ignored when using archival style transaction logging.

Attention: The minimum size is 192 MB (it ignores all lower values). If you start Domino with physically less disk space for TxL logs than 192 MB, Domino will identify this as a severe error (panic).

Note: Translog parameters are always overwritten by the values from the server document, so there is no point to changing them in the notes.ini directly. Domino has to create more than 65 MB of Archival TxL log before the Domino API reports the first complete archive log is complete and available for backup. Some third-party backup solutions use recent API additions allowing the active log to be flushed and a new log started to allow it to be backed up.

TxL logs are written in 64 KB chunks. The logs “segment” (next log file “extent”) at 64 MB. The logs are binary and created empty 64 MB long. When re-used, they are incrementally renamed and the contents cleared down. All of this helps performance and prevents disk fragmentation.

Never routinely delete TxL log files! The only purpose of TxL on iSeries is fast recovery after a crash and incremental backups.

Monitor the statistic Database.RM.SinceStartup.Critical.Log.Times. If this value is ever greater than 0, it suggests that at some point since server startup, the uncommitted transactions (those not hardened from virtual cached images to real disk images) in the (circular) log have nearly equalled the size of the log itself. Or it may indicate that new transactions are flowing through the log as fast, or faster, than the recovery manager can commit them. Since the recovery manager must not overwrite an uncommitted transaction in the logs, flushing the virtual database image to disk (hardening) gets a higher priority during these critical times. This then releases uncommitted transactions from the TxL logs to allow re-use of TxL log files. As a consequence of this, server performance suffers. To correct this, you can increase the log size, that is this parameter.

7.14.2 How queues relate to the NSF_Buffer_Pool_Size

A message queue is a queue that Notes uses to store certain tasks that are queued up (indexing, design updates, full text, agents, open handles for lookups). The Indexer is a classic example of a Domino “task” that works from a queue.

The following tasks access an Indexer queue:

- ▶ The Indexer accesses this queue to pull requests from it.
- ▶ Every “view update” request is added to this queue.
- ▶ Chronos (the Domino scheduler) adds requests for Full Text updating to this queue.

Note: Queue access is on a first-come, first-served basis; no task has priority over another task, and the requests are added to the queue in First In, First Out (FIFO) form.

So, the Indexer queue contains various requests for databases to be indexed from various sources. The Indexer reads the next request from this queue, removes it from the queue, and performs the indexing functions. Therefore, a single Indexer task works on a single database indexing request that it pulled from the queue. If a second request comes into the queue, the next indexer then removes that request from the queue and starts working on it. If both of these requests are for the same database, then the two tasks will attempt to work on the same database, assuming it is on different indexes.

It is statistically more likely, however, the two tasks will work on different databases. If the request is for a full text index update, this is a single index and multiple UPDATE processes cannot update the same full text index at the same time.

While multiple Indexer Tasks can update different view indexes within the same database at the same time, typically we don't recommend running large numbers of Indexers due to the fact that it may cause contention.

Tip: Loading multiple Indexer tasks on the server may or may not always be beneficial. It all depends on whether the multiple requests in the queue are for the SAME full text index on the SAME database.

When a message queue reaches its capacity, this is the error that occurs:

"Server Error: Insufficient Memory - Message Queue Pool is Full"

For the Indexer queue, you can check \$UPDATEQUEUE and see if it is the Indexer Queue that has reached its capacity. "Message Queue Pool" is Full for the Indexer task or tasks may indicate:

- ▶ We do not have sufficient indexers taking work from the Index Queue
- ▶ We are not servicing the pool fast enough (that is insufficient resources)
- ▶ The pool itself is full, causing the queue to reject further entries (pool too small)

This type of error message is seen most frequently on mail servers on which mail databases are full text indexed.

Here are some possible workarounds particularly for Mail Files:

- ▶ Load another Indexer task on the server to service the queue more frequently.
- ▶ Change the full-text update frequency from hourly to daily on the mail files.
- ▶ Limit the number of simultaneous full text Indexers with FullTextMultiProcess=.
- ▶ Prohibit all full text indexing (usually of user mail files) with Update_No_Fulltext=1.

A display of the Notes statistics may reveal the buffer pools maxed out. In Figure 7-17, the **sh stat database** command was entered on the server console (or remote console).

```

> SH STAT DATABASE
Database.BufferControlPool.Peak = 130,812
Database.BufferControlPool.Used = 40,814
Database.BufferPool.Maximum = 7194660
Database.BufferPool.Peak = 1728000
Database.BufferPool.PerCentReadsInBuffer = 98
Database.BufferPool.Reads = 4099
Database.BufferPool.Used = 1717134
Database.BufferPool.Writes = 747
Database.DbCache.CurrentEntries = 7
Database.DbCache.HighWaterMark = 37
Database.DbCache.Hits = 2404
Database.DbCache.InitialDbOpens = 3270
Database.DbCache.Lookups = 3256
Database.DbCache.MaxEntries = 25
Database.DbCache.OvercrowdingRejections = 239
Database.NIFPool.Peak = 130,812
Database.NIFPool.Used = 69,263
Database.NSFPool.Peak = 130,812
Database.NSFPool.Used = 20,812

```

Figure 7-17 The *sh stat* command examining the database buffer cache and pool

In a textbook example where there were major problems, the customer key values were:

- ▶ Database.BufferPool.Maximum = 74693652
- ▶ Database.BufferPool.Peak = 75110000
- ▶ Database.NIFPool.Peak = 2223804
- ▶ Database.NIFPool.Used = 4292080218
- ▶ Database.NSFPool.Peak = 1765962
- ▶ Database.NSFPool.Used = 4286027655

You should notice that:

- ▶ Database.BufferPool.Maximum < Database.BufferPool.Peak
- ▶ Database.NIFPool.Peak < Database.NIFPool.Used
- ▶ Database.NSFPool.Peak < Database.NSFPool.Used

Increasing the NSF_Buffer_Pool_Size in the notes.ini file may help to resolve the issue. However, the server may need to be monitored to see if the Database.BufferPool.Maximum still shows less than the Database.BufferPool.Peak or the “Message Queue Pool is Full” error continues. If this consistently happens, the server may be simply overloaded and need more memory or another resource to be upgraded.

NSF_DbCache_Maxentries

Each server uses a database cache to minimize the time lags involved with opening and closing databases on a server. When a user opens a database, Domino puts the database in the cache, where it remains for a period of time. While a database is in the cache, users can open and close the database quickly.

How databases are removed from the cache

Databases are removed from the cache through the process described here. An *ager thread* automatically “pushes” databases toward closure (by performing necessary writes, deallocating memory, and so forth) over a period of 15 to 20 minutes. Ideally, databases leave the cache in time to allow room for new entries, which are added when a database is closed. If not (the current number of entries exceeds NSF_DbCache_Maxentries), then one of the following actions occurs:

- ▶ If the current number of entries is less than $\text{NSF_DbCache_Maxentries} \times 1.5$, Domino adds a database entry to the cache and the ager accelerates to quickly get down to the NSF_DbCache_Maxentries setting.
- ▶ If the current number of entries is greater than or equal to $\text{NSF_DbCache_Maxentries} \times 1.5$, the database is closed normally instead of being put in the cache.

View_Rebuild_Dir

The View_Rebuild_Dir parameter changes the temporary work file directory where the UPDATE task keeps all of its work files:

View_Rebuild_Dir=<path to desired directory>

The presence of this parameter also causes all views including permuted or categorized views (for Domino Release 5.0.3 or later), to be rebuilt using the much faster *R5 optimized view rebuild*. This places all the temporary files (.DTF files) in the directory specified, instead of using the R4 default directory (which was the Domino data directory).

Note: Views rebuilt by R5 Optimized view rebuild can be much larger than the actual view size observed in the View Properties of the database and may require this temporary work file directory to have significant amounts of free space.

If insufficient space is available or it's an R4 database, the following warning appears and is logged:

“Warning: unable to use optimized view rebuild for <View-Name> due to insufficient disk space at <Dir>. Estimate may need <9999> million bytes for this view. Using standard rebuild instead”

In this case, the R5 Optimized view rebuild cannot be used and the behavior as in Domino R4 takes over.

We recommend that you use R5 Optimized view rebuild and suggest the following actions:

- ▶ Make certain system account has full rights to the temporary file directory named.
- ▶ Some platforms (not iSeries) require a trailing directory separator.
- ▶ Convert all databases to R5 to take full advantage of R5 Optimized view rebuild.
- ▶ For iSeries, also make sure the QNOTES user profile owns the directory or has full authority to it.
- ▶ This directory can be placed completely outside the data directory path. On iSeries, we recommend that you use a directory in its own ASP for optimal performance.

7.15 Controlling the details of Domino logging

Use Domino logging only if you suspect a performance problem, because extensive logging can seriously impact performance. You can use logging for trouble shooting. Always make a note of what logging you enable. Make sure that logging is switched *off* after the problem is resolved or that investigation ends.

SERVER_SHOW_PERFORMANCE

To display server performance events on the server console, use the following command:

```
show performance
```

This can be shortened to:

```
sh pe
```

This sets the notes.ini `Server_Show_Performance` parameter to a value of 1. The message “Server performance monitoring is now enabled” appears on the Domino server console and will run once a minute. If the notes.ini `Server_Show_Performance` parameter value is set to 0, server performance events will display in the Notes log only.

To stop server performance events from being displayed on the console, type **show performance** again and performance events will no longer be displayed on the console.

On the Server Console as well as in log.nsf, the concurrent transactions will appear as shown in the following example:

```
03:41:55 PM      41 Transactions/Minute,   5 Users
```

DEBUG_DISABLE_FAIRSEM

You may find that changing this undocumented parameter helps performance. In the current releases of Domino, the NIF-semaphores are type “FRWSem” (FairReadWriteSemaphores). This kind of semaphore allows either one writer or multiple readers (up to 48) at the same time. If there are more threads trying to read- or write-lock the semaphore, these are queued to be processed in the correct order (FIFO).

The semaphore-type for nif-semaphores may be changed by the notes.ini setting `DEBUG_DISABLE_FAIRSEM=1`. The semaphore type used instead will not queue the requests and, therefore, may result in lower performance. It is not recommended to be used unless there are serious performance issues and even then only with the advice of support.

Set the following parameters:

```
Debug_Capture_Timeout=1  
Debug_Show_Timeout=1
```

After you add these parameters, if semaphores are generated, they are captured in a file called SEMDEBUG.TXT in the Domino program directory. The output of the SEMDEBUG.TXT file looks like this example:

```
THREAD [02AB:0123] WAITING FOR FRWSEM 0x030B Collection semaphore (@0BE65A20)  
(R=0,W=1,WRITER=0080:015E,1STREADER=0000:0000) FOR 30000 ms
```

```
THREAD [01FE:0220] WAITING FOR FRWSEM 0x0244 open database semaphore (@00ECBDD2)  
(R=0,W=1,WRITER=0080:015E,1STREADER=0000:0000) FOR 30000 ms
```

If you see a lot of FRWSem and semaphores are timing out, disabling fair semaphores may be appropriate. Again, the underlying cause of semaphore time-out should be investigated with support, rather than masking it by removing the queueing.

7.16 iSeries environment variable settings

iSeries environment variables are used to control the following specific attributes of the Domino server. Changes to these variable are activated after the server is terminated and restarted again. You add new ones with the Add Environment Variable (ADDENVVAR) command. By default, environment variables only persist for the duration of the job that submitted the ADDENVVAR command, unless you specify the parameter LEVEL(*SYS).

Notes_AS400_CONSOLE_ENTRIES

This is the size of the console file that displays the status messages when you use the Work with Domino Console (WRKDOMCSL) or Display Domino Console (DSPDOMCSL) command. As a log file increases in size, the response time on any commands on the Domino console also increases.

Notes_SHARED_POOLSIZE

The default value for this was 1 MB at Release 4.6b. It was increased to 12 MB for Release 4.6.2. The current default is 12 MB. You do *not* need to set this variable for Domino R5.

Other environment variables used by Domino for iSeries are:

- ▶ **FaultRecovery:** Defines whether QNNINSTS will attempt to automatically restart the server after a task ends abnormally.
- ▶ **AS400_DOMSVR_RESTART:** Defines the maximum number of restarts.

See the *Domino for AS/400 Help* database help/as400hlp.nsf on your Domino server for more information on environment variables.

7.17 Domino for iSeries specific notes.ini parameters

The following parameters are not necessarily unique to Domino for iSeries, but are important for Domino when running iSeries servers.

NSF_HOOKS

This is a variable in the notes.ini file that is usually only used by the directory synchronization task and when set the entry usually reads NSF_HOOKS=QNNDIHK.

Theoretically, there could be other entries on that line (that is hooks used by other applications), so just remove the QNNDIHK entry and the associated comma:

NSF_HOOKS=my_application_1,QNNDIHK,my_application_2

Important: If you are *not* using directory synchronization, you should perform each of the following actions:

- ▶ Remove the “QNNDIHK” entry from the “NSF_HOOKS=” line in the notes.ini file.
- ▶ Remove the “QNNINADD” entry from the “SERVERTASKS=” line in the notes.ini file.
- ▶ Issue the following O/400 CL command to end directory synchronization:
call qnotesint/qnndiend

One of the processes during the installation of the Domino server with Option 1 selected creates an autostart job entry in subsystem description QSYSWRK. QNNDIRQS is started after an IPL as a result of the autostart job entry. The autostart job entry can be verified and removed using the following commands from an OS/400 command line.

1. Verify that the autostart job entry exists using Work with Subsystem Descriptions screen
 - a. Enter 5 on the option line next to QSYSWRK.
 - b. Select option 3 (Autostart job entries).
 - c. The Display Autostart Job Entries screen may display job QNNDISTJ in library QSYS.
 - d. If it exists, the QNNDISTJ entry needs to be removed (see next step).

2. Remove the Autostart job entry from the command line with the command:

```
RMVAJE SBSD(QSYS/QSYSWRK) JOB(QNNDISTJ)
```

Note: The above action does not take effect until the subsystem is ended (ENDSBS QSYSWRK) and restarted (STRSBS QSYSWRK).

EnableJavaAgents

Another item to carefully scrutinize is the EnableJavaAgents variable in the notes.ini file. This variable enables or disables the ability to run Java agents in your Domino server. If the AS/400 Developer Kit for Java (5769-JV1 or 5722-JV1) is installed on your iSeries, the Domino Agent Manager (AMGR) and HTTP server automatically support running Java agents.

You can disable Java agents by adding the following line to the notes.ini file for the Domino server:

```
EnableJavaAgents=0
```

MAILMAXTHREADS

Controls the maximum number of threads that the mail router job can create. The setting of this variable is very important for mail hub servers. To appropriately set this variable, we recommend that you use the **show server** command at the console and check for pending mail. Increase MAILMAXTHREADS by 1 until pending mail shows 0 or the desired level of performance is reached.

ServerTasks

The SERVETASKS variable controls which tasks are started when a Domino server is brought up. By limiting the number of tasks that start, you can improve the performance of that Domino server. For example, if you partitioned your Domino servers to handle different Notes functions, only start the tasks that are needed for a particular server.

Note: Be careful about removing the UPDATE task. Removing the UPDATE task can cause serious functional problems on your Domino server if you have an application that depends on view and index updates that occurs “on the fly”.

Here are some suggestions for which tasks you might consider removing from the ServerTasks entry in notes.ini. It is likely that very few of these are already in your notes.ini since many are not there by default. However, the defaults have changed with various releases. They also change depending on the options chosen when you configured your iSeries Domino servers. The tasks ICM, HTTP, IIOP, IMAP, POP3, LDAP, QNNINADD, DECS, LDAP, SMTPMTA, and NNTP are all configurable at the time a server is installed.

Note: The ServerTasks variable is preserved when upgrading, so newer functions may not have been added automatically.

Table 7-3 lists the majority of the server tasks and summarizes their functions with comments.

Table 7-3 Domino for iSeries server tasks and their descriptions

Task command to run	Task description
Administration Process AdminP	Automates a variety of administrative tasks. You should not remove this entry.
Agent manager AMgr	Runs agents on one or more databases. Generally, you should not remove this. Doing so disables all scheduled agents on the server.
Billing	Billing Collects all generated billing information. If you are not using the Billing function, you may remove this.
Calendar Connector Calconn	Processes requests for free-time information from another server. If you do not use Calendaring and Scheduling or if you use it <i>only</i> on one server, you may remove this. Doing so disables free-time lookups to other servers.
Cluster Database Directory Manager Cldbdir	Updates the cluster database directory and manages databases with cluster-specific attributes. If you have not implemented clustering or if this server is not a member of any Domino cluster, you may remove this.
Cluster Replicator Clrepl	Performs database replication in a cluster. If you have not implemented clustering or if this server is not a member of any Domino cluster, you may remove this.
DIIOP	DIIOP allows Domino and the browser client to use the Domino Object Request Broker (ORB) server program. If you have written or purchased Domino applications that contain Java pallets utilizing the Domino Java Classes, you need to enable this. It is not enabled by default. If you do not run the HTTP task, you do not need this.
HTTP Server	HTTP enables a Domino server to act as a Web server so browser clients can access databases on the server. If you do not wish to allow any Web browser access to the Domino server, this may be removed.
IMAP Server	IMAP enables a Domino server to act as a maildrop for IMAP clients. If you have no IMAP clients, you may remove this.
IMAPMaxSessions	R5.0.3 specifies the maximum number of sessions that will be allowed in the IMAP server. Default=0 (or unspecified) = No Session Limit
ISpy RunJava	ISpy sends server and mail probes and stores the statistics. It is used primarily for troubleshooting
LDAP Server	LDAP enables a Domino server to provide LDAP directory services to LDAP clients. Only needed if you support LDAP queries. These usually come from POP3 and IMAP e-mail clients.

Task command to run	Task description
MTC	MTC reads log files produced by the router and writes summary data about message traffic to a database for message tracking purposes. Only needed if you are using Mail Tracking.
POP3 Server	POP3 enables a Domino server to act as a maildrop for POP3 clients. If you have no POP3 clients, you may remove this.
NSF_HOOKS=QNNDIHK and SERVERTASKS=QNNINADD	This nsf_hooks and servertask pair synchronize the Domino Directory with the iSeries Distribution Directory (SDD). If you are <i>not</i> using directory sync you may disable it. For details, see “NSF_HOOKS” on page 224.
Replicator Replica	Replicates databases with other servers. Do not remove.
Router Router	Routes mail to other servers. Do not remove.
Schedule manager Sched	Returns meeting times and dates and available invitees. If you are not using Calendaring and Scheduling at all, you may remove this.
Stats	Stats generates statistics for a remote server on demand. Do not remove.

Note: This is not a complete list of Domino tasks. See the Domino Help Administration database for descriptions of other Domino server tasks.

A typical R5 default ServerTasks variable looks like this:

```
ServerTasks=Billing,Replica,Router,Update,Stats,AMgr,Adminp,Sched,CalConn,Event,QNNINADD
```

7.18 CORBA and Java servlets

For best CORBA and Java servlet performance, make sure a JVAPGM is associated with any JAR files you are using in your servlet. For example, you would typically use /qibm/proddata/lotus/notes/SHARED/NSCOW.jar for a WebSphere servlet to access Domino via CORBA. NCSOW.jar doesn't ship with a JVAPGM so you have to create one if you are going to use it on the iSeries server. To do this, you can use the Create Java Program (CRTJVAPGM) command. Enter the following command on any OS/400 command line:

```
CRTJVAPGM CLSF('/qibm/proddata/lotus/notes/SHARED/NSCOW.jar') OPTIMIZE(30)
```

You can check if one exists, or what its optimization level is by using Display Java Program (DSPJVAPGM) command on the IFS file:

```
DSPJVAPGM CLSF('/qibm/proddata/lotus/notes/SHARED/NSCOW.jar')
```

When using Java agents that reference external Java packages, do not put the entire JAR file into your agent. JAR files stored in an .NSF file with an agent cannot take advantage of having a Direct Executable JVAPGM associated with them. For the NOTES.JAR, you do not need this. It is in the CLASSPATH for your agent when it runs already.

For other third-party packages, such as the IBM Toolbox for Java (previously known as Java Toolbox for AS/400; JT400.JAR), add these to the CLASSPATH via the JavaUserClasses entry in notes.ini instead of putting them directly into the agent. Make sure they have a JVAPGM created for them (using CRTJVAPGM) and that QNOTES has authority to use them. For example, to use the JT400.JAR file that ships with 5769-JC1 (IBM Toolbox for Java) and another .JAR from Whizbang Corp called WHIZBANG.JAR, you'd enter this in the notes.ini file:

```
JavaUserClasses=/qibm/proddata/http/public/jt400/lib/jt400.jar:/whizbang/lib/whizbang
```



Understanding the Domino server jobs

This chapter discusses the Domino server tasks. It provides insight into each OS/400 job that represents one of those tasks, how these jobs relate to the implementation of Domino for iSeries, and what you can do to tune your server environment.

The key to performance tune the server is to understand the relationships between what appear to be traditional iSeries jobs and Domino jobs. Due to the scalability of the iSeries server, you then determine if creating another server and subsystem for a specific server job within the iSeries would improve the performance of the entire environment. This chapter also offers some best practices that can provide a framework for your server environment.

For fully detailed information about Domino tuning, see Chapter 7, “Tuning Lotus Domino for better performance on iSeries” on page 183.

Important: In this chapter, when we discuss Domino server jobs, it is the same as if we use the term *Domino server tasks*. This may be confusing. However, if you see it from an iSeries point of view, then it is *server jobs*, and if you see it from a Domino perspective, it is *server tasks*.

Domino for iSeries allows flexibility in the management of Domino server jobs. Only the iSeries server platform allows you to manage all jobs related to one Domino in one entity: a subsystem. This provides increased flexibility during the implementation of Domino servers.

Only through in-depth understanding of which server tasks are important to your organization, you can manage the Domino server environment. Once you identify the critical server jobs that need to be used for the organization, you are in a position to analyze such items as:

- ▶ Number of users
- ▶ Transaction rates per minute
- ▶ Response time
- ▶ Disk requirements

The following sections detail an instance of a server:

- ▶ Section 8.2, “Domino server jobs (always necessary)” on page 232, covers server tasks that need to run on the Domino server. Those jobs are started by default when you configure a new Domino server.
- ▶ Section 8.3, “Domino server jobs (often used)” on page 245, covers advanced jobs that may be needed in your environment. However, exploiting the function sets that are running under these jobs will directly and positively impact the performance of your entire Domino environment.
- ▶ Section 8.4, “Domino server jobs (very likely not used)” on page 253, describes those tasks that may not be needed on most Domino servers. Since many tasks are started unnecessarily by default, it is worth verifying whether you need certain functions.
- ▶ Section 8.5, “Domino server database housekeeping jobs” on page 256, covers additional jobs that are run in a background environment. That is, they are started typically by the Domino system administrator or an agent and run only for a specified period of time.

Note: Some of the tasks are started by default, changing the `ServerTasks=` line in `notes.ini`. A typical default content of that parameter may be:

```
ServerTasks=Replica,Router,Update,Stats,AMgr,Adminp,Sched,CalConn,Event
```

Others will be added to start based on the selection made when configuring the Domino server or manually to `notes.ini`.

Each section provides the following information for that particular job running in the Domino server subsystem:

- ▶ Description of Domino server functions running under that job
- ▶ Potential performance impact of Domino servers function under that job
- ▶ Suggested steps to performance tune that job or function set

8.1 Domino server jobs

Domino server jobs run under the subsystem you defined when you configured the server. If you did not specify a name for the subsystem, the name `DOMINO nn` will be used, where nn is 01 for the first¹ server you configure on your iSeries. It will be increased by one for each new partitioned server on your system.

To see the all the server jobs related to one Domino server (therefore running under one subsystem), you can either use Operations Navigator or enter the `WRKDOMSVR` or `WRKACTJOB` CL commands on an OS/400 command line.

Operations Navigator provides an intuitive, graphical user interface for iSeries operators and administrators. To manage you Domino for iSeries servers, perform the following steps:

1. Start Operations Navigator on your Windows workstation.
2. Expand the tree under **My iSeries Connections** by single clicking the plus sign (+).
3. Expand the tree under the name of your iSeries server. When prompted, enter your user ID and password to access that system.
4. Expand the tree under **Network**.
5. Expand the tree under **Servers**.

¹ This should not be confused with the parameters `*FIRST` and `*ADD` in the `Configure Domino Server (CFGDOMSVR)` command. These refer to the first or additional servers within the Domino organization or domain.

- Right-click **Domino** and select **Domino jobs** to see the jobs of *all* Domino servers running on your iSeries as shown in Figure 8-1.

Job name	User	Job type	Job status	Time entered	Date entered	Subsystem
Qnnndirqs	QNOTES	Batch	Active - QNNDIRQS program	18:20:13	06.12.01	QSYSWRK
Qnninsts	QNOTES	Batch	Active - QNNINSTS program	05:53:33	11.12.01	DOMINO01
Server	QNOTES	Batch	Active - SERVER program	05:53:35	11.12.01	DOMINO01
Logasio	QNOTES	Batch	Active - LOGASIO program	05:53:40	11.12.01	DOMINO01
Cladmin	QNOTES	Batch	Active - CLADMIN program	05:54:25	11.12.01	DOMINO01
Qnninbrm	QNOTES	Batch	Active - QNNINBRM program	05:54:26	11.12.01	DOMINO01
Billing	QNOTES	Batch	Active - BILLING program	05:54:32	11.12.01	DOMINO01
Replica	QNOTES	Batch	Active - REPLICA program	05:54:37	11.12.01	DOMINO01
Router	QNOTES	Batch	Active - ROUTER program	05:54:43	11.12.01	DOMINO01
Update	QNOTES	Batch	Active - UPDATE program	05:54:49	11.12.01	DOMINO01
Amgr	QNOTES	Batch	Active - AMGR program	05:54:54	11.12.01	DOMINO01
Amgr	QNOTES	Batch	Active - AMGR program	05:54:56	11.12.01	DOMINO01
Adminp	QNOTES	Batch	Active - ADMINP program	05:55:00	11.12.01	DOMINO01
Clbdir	QNOTES	Batch	Active - CLBDIR program	05:55:02	11.12.01	DOMINO01
Clrepl	QNOTES	Batch	Active - CLREPL program	05:55:04	11.12.01	DOMINO01
Sched	QNOTES	Batch	Active - SCHED program	05:55:05	11.12.01	DOMINO01
Calconn	QNOTES	Batch	Active - CALCONN program	05:55:11	11.12.01	DOMINO01
Event	QNOTES	Batch	Active - EVENT program	05:55:18	11.12.01	DOMINO01
Decs	QNOTES	Batch	Active - DECS program	05:55:23	11.12.01	DOMINO01
Collect	QNOTES	Batch	Active - COLLECT program	05:55:29	11.12.01	DOMINO01
Clrepl	QNOTES	Batch	Active - CLREPL program	05:55:45	11.12.01	DOMINO01
Qnninsts	QNOTES	Batch	Active - QNNINSTS program	19:48:16	11.12.01	QDOMINO1
Http	QNOTES	Batch	Active - HTTP program	19:48:17	11.12.01	QDOMINO1

Figure 8-1 Active Domino jobs shown in Operations Navigator

You may customize the display by selecting **Options-> Columns** in the menu bar to add, re-sequence, or delete columns.

If you prefer using CL commands, you may enter one of the following commands on any OS/400 command line:

- ▶ The Work With Domino Servers (WRKDOMSVR) command shows all Domino servers on your iSeries. Then, select the server and type **9** (Work server jobs) in the option field next to the name of the server.
- ▶ Alternatively, you can enter the Work with Active Jobs command from an OS/400 command line:

```
WRKACTJOB SBS(MAILSVR)
```

Specify the name of the subsystem with the SBS parameter.

Figure 8-2 shows an example of server jobs that run under a subsystem named MAILSVR. Note that this is not a complete list of jobs that can run under a Domino server subsystem.

Work with Active Jobs						AS06
						11/12/01 09:58:04
CPU %:	9.8	Elapsed time:	00:10:28	Active jobs:	296	
Type options, press Enter.						
2=Change 3=Hold 4=End 5=Work with 6=Release 7=Display message						
8=Work with spooled files 13=Disconnect ...						
Opt	Subsystem/Job	User	Type	CPU %	Function	Status
	MAILSVR	QSYS	SBS	.0		DEQW
	ADMINP	QNOTES	BCI	.0	PGM-ADMINP	SELW
	AMGR	QNOTES	BCI	.0	PGM-AMGR	SELW
	BILLING	QNOTES	BCI	.0	PGM-BILLING	SELW
	CALCONN	QNOTES	BCI	.0	PGM-CALCONN	CNDW
	COLLECT	QNOTES	BCI	.0	PGM-COLLECT	SELW
	CLADMIN	QNOTES	BCI	.0	PGM-CLADMIN	SELW
	CLDBDIR	QNOTES	BCI	.0	PGM-CLDBDIR	CNDW
	CLREPL	QNOTES	BCI	.0	PGM-CLREPL	SELW
	EVENT	QNOTES	BCI	.3	PGM-EVENT	SELW
	HTTP	QNOTES	BCI	.0	PGM-HTTP	SELW
	LOGASIO	QNOTES	BCI	.0	PGM-LOGASIO	CNDW
	QNNINSTS	QNOTES	BCH	.0	PGM-QNNINSTS	EVTW
	REPLICA	QNOTES	BCI	.0	PGM-REPLICA	SELW
	ROUTER	QNOTES	BCI	.0	PGM-ROUTER	SELW
	SCHED	QNOTES	BCI	.0	PGM-SCHED	SELW
	SERVER	QNOTES	BCI	.5	PGM-SERVER	SELW
	SMTP	QNOTES	BCI	.0	PGM-SMTP	SELW
	STATS	QNOTES	BCI	.0	PGM-STATS	SELW
	UPDATE	QNOTES	BCI	.0	PGM-UPDATE	SELW

Figure 8-2 WRKACTJOB SBS(MAILSVR): Domino server jobs

8.2 Domino server jobs (always necessary)

The server tasks described in this section should normally not be removed from the ServerTasks statement in the notes.ini file. They are started by default on every server and should stay active.

Note: When a new server is created, some jobs start by default that might not be needed at all. These kind of jobs are discussed in 8.4, “Domino server jobs (very likely not used)” on page 253.

8.2.1 ADMINP: Administration process

The administration process is a server task called ADMINP that automates many routine administrative tasks. You choose an action, and the administration process completes it for you. The administration process can perform these administrative tasks:

- ▶ Create mail files during setup
- ▶ Upgrade users and servers to hierarchical naming
- ▶ Rename users
- ▶ Recertify users and servers
- ▶ Delete users, servers, and groups

- ▶ Add resources to and delete them from the Domino Directory when these actions are initiated in the Resource Reservations database
- ▶ Set the Master Address Book field in the Domino Directory to enable directory assistance
- ▶ Create replicas of multiple databases
- ▶ Enable password checking during authentication
- ▶ Add servers and remove servers to and from a cluster
- ▶ Move databases from a cluster server
- ▶ Route mail to other servers

In addition to these tasks that you initiate, the administration process automatically copies a public key of the server to its server record when you start the server and the public key is not there already. It also places the server build number in the server document. The administration process can do these two tasks in a flat or hierarchical environment.

Potential performance impact

The ADMINP server job can be very CPU intensive. The tasks performed by ADMINP are enormous when compared to other tasks in the Domino environment. Due to the various and numerous tasks involved, any changes to ADMINP should be done with caution. It is possible, due to the exhausting nature of the tasks completed by ADMINP that your Domino environment could be negatively impacted by its activities. Let us talk about some of the big tasks that may very well negatively impact your Domino server performance.

There are two main activities that, relatively speaking, have a more devastating impact on your Domino server environment when they are started by ADMINP:

▶ Recertification of user IDs

This is mostly due to the impact on the Domino Directory (Name and Address Book) for the organization. Upon recertification, the Domino Directory must be updated with the new information regarding these IDs. As a result, a recertification task and a re-indexing task are executed against the Domino Directory. Therefore, the overhead in accomplishing this task can be enormous.

▶ Database moves in a clustering environment

This is more of a function of the number of databases that are being copied across the cluster. If you have a limited number of databases, the impact is less than if you have a large number of databases.

Stopping ADMINP

There may be cases where a server task, such as ADMINP, has been processing requests for much longer than expected. This impacts the server's performance. The server console command `te11 adminp q` brings the tasks down while doing the appropriate cleanup. Do not use the OS/400 End Job (ENDJOB) command.

For more information, refer to the site <http://www-4.ibm.com/software/support/> and search for document **180515**.

Suggested performance tuning

When analyzing the impact of ADMINP on your Domino server performance, consider the suggestions described here.

The exposure to database moves can be reduced by implementing this particular task outside of your normal business hours. There are several ways to initiate the ADMINP job. Please refer to the *Domino Administration Help Database* that comes with the Domino server software. One way is to schedule ADMINP functions:

- Immediate
- Scheduled
- Nightly
- Weekly

The tasks completed by ADMINP can have a negative effect on users during normal business hours. Therefore, you may want to balance the organizational requirements for the ADMINP tasks against running these tasks outside of the normal business hours of your user community.

There are several factors that affect the time to complete many of the tasks provided in ADMINP. For more information, refer to the site <http://www-4.ibm.com/software/support/> and search on document **151808**.

Figure 8-3 shows the Administration process (ADMINP) of the server document.

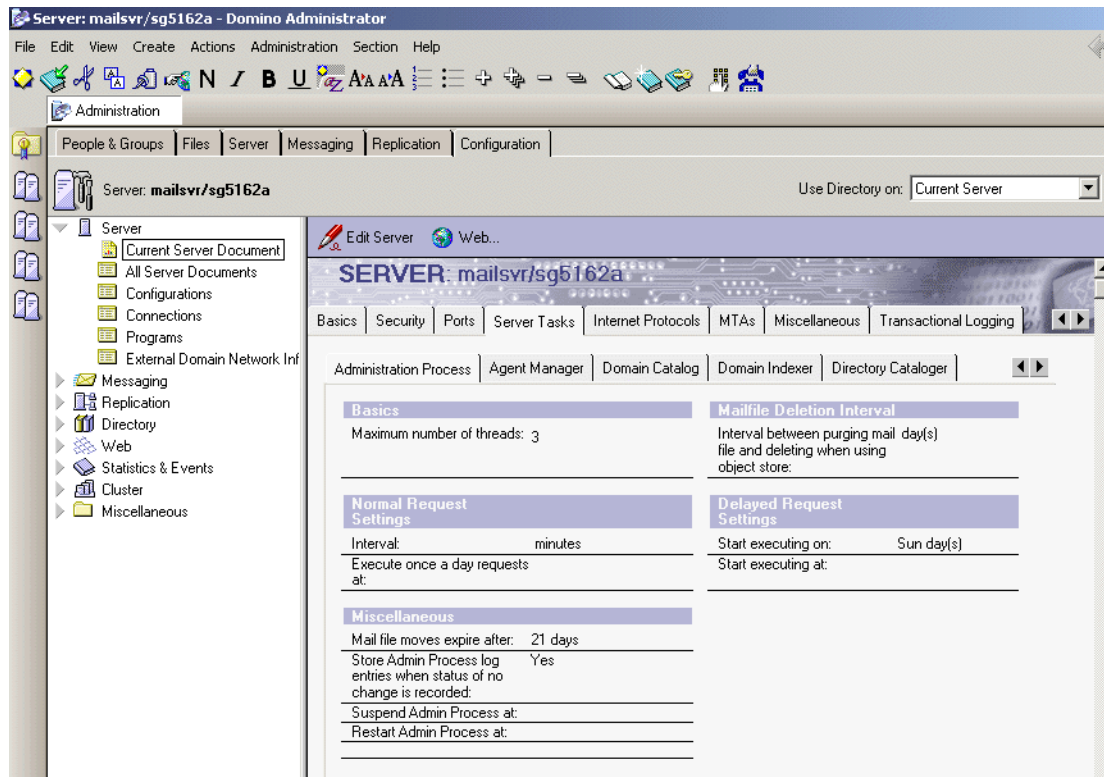


Figure 8-3 Administration process section of the Server Document in the Domino Directory

To understand the impact of the ADMINP task on the entire CPU utilization of iSeries server, you may log the platform dependent statistics in the statrep.nsf database. See Figure 5-26 on page 155 for an example and 5.3, “Combining performance data from Domino and iSeries” on page 152, for information on setting it up.

8.2.2 AMGR: Agent manager

Agents let Notes users automate tasks. Users are provided with a set of predefined agents. For example, an agent may automatically send a response to all incoming e-mail, indicating that a user is away on vacation or business. Users can use the Notes Agent Builder to create simple agents, formula agents, or LotusScript agents. There are two types of agents:

- **Personal agents:** For use only by the user who creates the agent
- **Public agents:** Created by a database designer for use by specified users

Before the Agent Manager task runs a LotusScript agent, it performs the following security-related steps:

1. Validates the signature on the agent note. If the signature is missing, the agent cannot run.
2. Checks the server document in the Domino Directory to determine whether the user can run restricted and unrestricted agents.
3. Checks for any API restrictions.
4. Begins agent execution.

Identifying potential performance impacts of AMGR

Because of what the AMGR does during an instance of any agents, the overhead when running agents can be an exposure.

The real exposure to AMGR predominantly comes from database design personnel. The number of agents and when they execute against a database and application can significantly impact Domino server performance. Due to the uniqueness of the database and application design, it is impossible to predict what affect agents will have in your environment. However, they will have a negative impact on server performance. To analyze the potential impact, issue the following command and note the results:

```
COMMAND SENT: show stat agent
Agent.Daily.AccessDenials = 0
Agent.Daily.ScheduledRuns = 3
Agent.Daily.TriggeredRuns = 0
Agent.Daily.UnsuccessfulRuns = 0
Agent.Daily.UsedRunTime = 208 Seconds
Agent.Hourly.AccessDenials = 0
Agent.Hourly.ScheduledRuns = 0
Agent.Hourly.TriggeredRuns = 0
Agent.Hourly.UnsuccessfulRuns = 0
Agent.Hourly.UsedRunTime = 0 Seconds
```

Logging agent activities

You can also log information about activities performed by AMGR. The *Log_AgentManager* setting in notes.ini provides you with a subset of the debugging information that *Debug_AMgr* generates. This option provides less output, but it has a smaller impact on performance. Some people keep the *Log_AgentManager* setting turned on even when there are no problems, just to have additional information in the log. The following values can be specified for the *Log_AgentManager* statement:

- 0** To not show logging
- 1** To show partial and complete successes
- 2** To show complete successes

A more powerful and, therefore, much more overhead producing function is *Debug_AMgr*. You can debug the agents by adding the following variable to notes.ini:

```
DEBUG_AMGR=n
```

Here *n* can have the following values:

- r** Agent execution
- l** Agent loading
- m** Agent memory warning
- p** Agent performance stats

- e Events
- s Schedule
- c Agent control parameters display
- v Verbose
- * Enables everything

If you have both notes.ini variables specified, Debug_AMgr settings take precedence.

Suggested performance tuning

As the administrator, you set up the Agent Manager to control who has the authority to execute agents on a specified server. By limiting when agents can run and how many can run at one time, you control server resources. This requires prioritizing those agents that must run during normal business hours to meet your organizational requirements and those that can be deferred until after hours.

AMGR lets you control when agents run. By default, daytime is defined as 8 a.m. to 8 p.m., and nighttime is from 8 p.m. to 8 a.m. Also, by default, more system resources are allocated to the Agent Manager at night. If you are running complex LotusScript agents, you should probably schedule them to run at night, when system demand is lower and the agents run faster. Remember that your primary objective is your Service Level Agreement (SLA) to your end-user community. Agents can be CPU bound.

Figure 8-4 shows the Agent Manager section of the Server document.

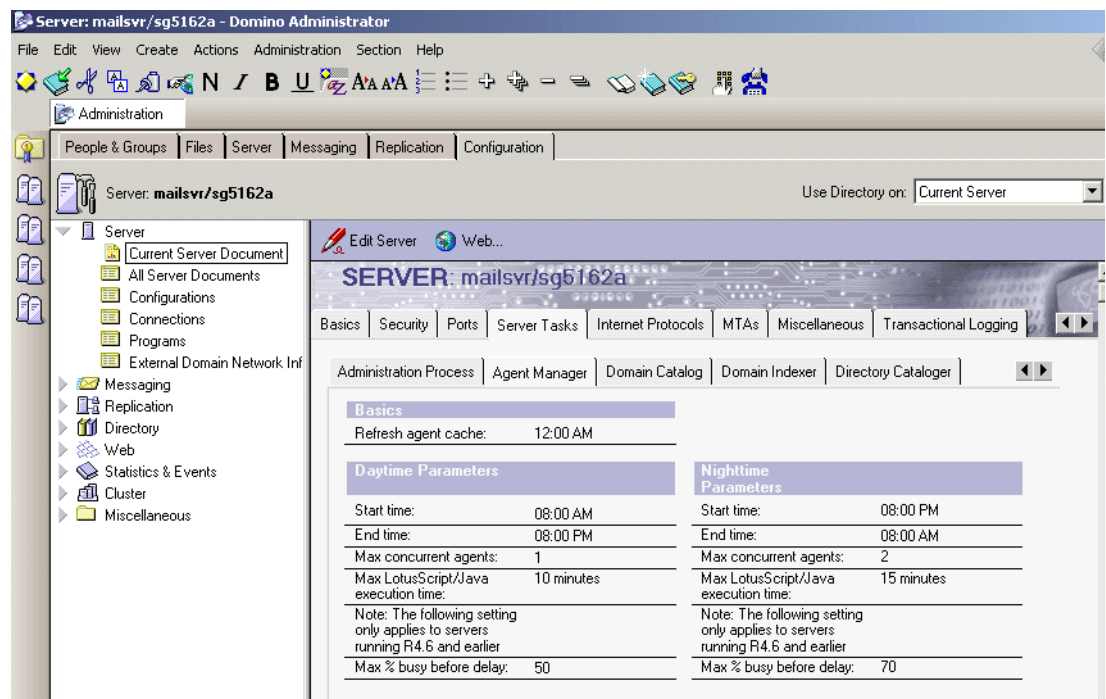


Figure 8-4 Agent Manager section of the Server document in the Domino Directory

Figure 8-5 show an example of the notes.ini server configuration document - Agent Manager.

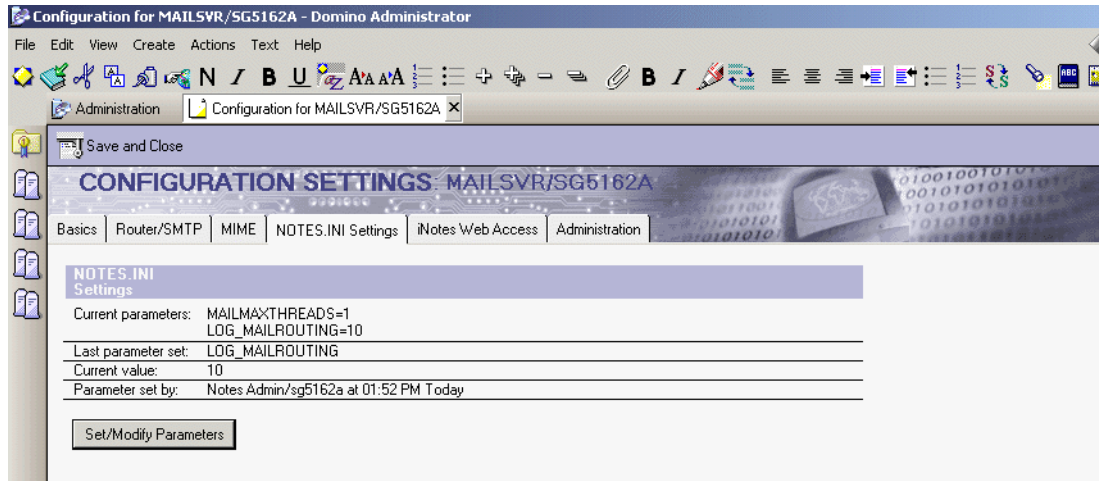


Figure 8-5 Notes.ini server configuration document - Agent Manager

Controlling how many concurrent agents are running

You can relieve a heavily loaded Agent Manager by allowing agents to run concurrently. To do this, modify the “Max concurrent agents” field in the Server Tasks/Agent Manager section of the Server document. Values greater than 1 allow more than one agent to run at the same time. Valid values are 1 through 10. Default values are 1 for daytime and 2 for nighttime. However, we strongly recommend that you carefully study the impact of multiple agent managers running carefully.

It is very unlikely to gain improvements by concurrently running multiple instances of agent manager on systems with less than three processors, unless you have agents with very long waits on I/O.

To understand the impact of the AMGR task or tasks on the entire CPU utilization of the iSeries server, you may log the platform dependent statistics in the statrep.nsf database. See Figure 5-26 on page 155 for an example and 5.3, “Combining performance data from Domino and iSeries” on page 152, for information to set it up. The statrep.nsf can store the CPU utilization of up to four instances of AMGR.

To see a snapshot of the Agent Manager status, including the number of Agent Executives currently running, enter the following command at the server console:

```
tell amgr status
```

Improving Agent Manager performance

The Agent Manager controls when agents run on a server. Every time an agent runs, it uses server resources. To control when scheduled and event-triggered agents run, you specify settings in the Server document and in the notes.ini file. Customizing when agents run may conserve server resources, but it may also delay when agents run.

Controlling how often Agent Manager runs agents

The following notes.ini settings affect how often the Agent Manager executes agents. In general, the more frequently agents run, the sooner they perform their tasks. Running agents more frequently, however, may increase demand on server resources and adversely affect overall system performance.

► **AMgr_DocUpdateAgentMinInterval**

This setting specifies the minimum elapsed time, in minutes, between executions of the same document update-triggered agent. This lets you control the time interval between executions of a given agent. Default is 30 minutes. A longer interval can result in the agent running less often, reducing server demand. If document update events are infrequent, you can reduce the delay. (Note that setting this and other Agent Manager variables to zero does not completely eliminate the delay; a built-in delay will always exist.)

► **AMgr_DocUpdate EventDelay**

This setting specifies the delay time, in minutes, that the Agent Manager schedules a document update-triggered agent after a document update event. The default is 5 minutes. The delay time ensures the agent runs no more often than the specified interval, regardless of how frequently document update events occur.

When the agent executes, it also processes all additional events (if any) that occurred during the interval. A longer interval results in the agent running less often, therefore, reducing demand for server time. If document update events are infrequent, however, you can reduce the delay to ensure the agent runs soon after the event occurs.

► **AMgr_NewMailAgentMinInterval**

This setting specifies the minimum elapsed time, in minutes, between executions of the same new mail-triggered agent. The default is 0 (no interval between executions). Similar to AMgr_DocUpdateAgentMinInterval, entering an interval can result in the agent running less frequently.

► **AMgr_NewMailEventDelay**

This setting specifies the time (in minutes) that the Agent Manager delays before scheduling a new mail-triggered agent after new mail is delivered. The default is 1 minute. Similar to AMgr_DocUpdateEventDelay, the delay time ensures the agent runs no more often than the specified interval.

► **DominoAsynchronizeAgents**

This setting specifies whether Web agents triggered by browser clients can run at the same time (asynchronously). The default is zero (only one agent can run at a time). Set this to 1 to allow multiple agents to run simultaneously. This can result in faster execution of agents. However, a high number of agents executing at the same time can slow overall system performance.

Controlling how quickly the Agent Manager queues agents

The Agent Manager periodically checks to see if it has any new agents that it needs to schedule. These notes.ini settings control how quickly an agent gets into the schedule queue:

► **AMgr_SchedulingInterval**

This setting specifies a delay (in minutes) between running the Agent Manager's scheduler. Valid values are 1 minute to 60 minutes. The default value is 1 minute.

► **AMgr_UntriggeredMailInterval**

This setting specifies a delay (in minutes) between running the Agent Manager's check for untriggered mail. Valid values are 1 minute to 1440 minutes (the number of minutes in a day). The default value is 60 minutes.

Controlling when the Agent Manager runs agents

When you create or modify an event-triggered agent, the Agent Manager schedules it to run immediately. This ensures the agent can quickly process new documents. These notes.ini settings let you specify a time interval between subsequent runnings of the agent. This can prevent repeated runnings of the agent, for example, because of a rapid series of triggering events.

► **AMgr_DocUpdateEventDelay**

This setting specifies a delay of execution (in minutes) that the Agent Manager includes in the schedule for a document update-triggered agent after a document update event. The default is 5 minutes.

► **AMgr_NewMailEventDelay**

This setting specifies a delay of execution (in minutes) that the Agent Manager includes in the schedule for a new mail-triggered agent after a new mail message is delivered. The default 1 minute.

► **AMgr_DocUpdateAgentMinInterval**

This setting specifies the minimum elapsed time (in minutes) between execution of the same document update-triggered agent. The default is 30 minutes.

► **AMgr_NewMailAgentMinInterval**

This setting specifies the minimum elapsed time (in minutes) between execution of the same new mail-triggered agent. The default is 0.

Monitoring the load on the Agent Manager

Domino Release 4.6 and earlier releases include the field “Max % busy before delay” in the Server document. This field limits the percentage of time the Agent Manager can use to run agents. When this limit is exceeded, the Agent Manager delays agent execution. To improve Agent Manager performance, Domino Release 5 does not include this limitation. You can, however, still use this field for Domino Release 4.6 and earlier releases.

If your server attempts to schedule agents at a rate faster than the Agent Manager can run them, the message “AMgr: Agent scheduling is paused” appears on the console. The Agent Manager will not schedule any new agents until the server processes some agents that are already scheduled. Therefore, the running of new agents may be slightly delayed.

8.2.3 QNNINSTS: The ‘Watchdog’

After the Start Domino Server (STRDOMSVR) command has been issued² and the subsystem is started, a job named QNNINSTS is submitted, which calls program QNNINSTS. This program in turn starts all server tasks and monitors whether any of them end abnormally. If one task fails, QNNINSTS tries to end all remaining tasks gracefully and restarts the server. Note, most of the jobs running under a Domino subsystem relate to a Domino server task. This does not apply to QNNINSTS.

While QNNINSTS is the first server job to be started, it is always the last to be ended. However, it typically does not reflect consumption of system resources. The job log stores all error messages logged on the console at startup.

Same as for all other Domino jobs, you should not end QNNINSTS using the End Job (ENDJOB) command. Use the End Domino Server (ENDOMSVR) command instead.

² In OS/400 V5R1 or later, this can be done automatically by the TCP/IP autostart functions when the AUTOSTART(*YES) parameter is used with the Configure or Change Domino Server (CFGDOMSVR or CHGDOMSVR) command. TCP/IP can be started automatically after Initial Program Load (IPL) by using the CHGIPLA STRTCP(*YES) command.

8.2.4 ROUTER: The mail router

Router is the Domino server task that moves mail messages between the sender's mail file and the recipient's mail file. *Mail routing* is the process of moving a message along a path of servers until the Mail Router on a server locates the mail file for the recipient. To help determine the path, the Mail Router on every server loads a routing table based on the information supplied in the Domino Directory.

When you register users, each user automatically gets a personal mail file. This mail file resides on the user's home server. When a user sends a mail message, the server's Mail Router takes on the job of delivering the message to the next stop on the route to the recipient.

The Mail Router on each server works with two databases located on each server:

- ▶ The mailbox database (mail.box), which stores mail pending delivery and routing
- ▶ The Domino Directory (names.nsf), which is a directory of users, servers, and connections between servers in the same Domino domain and between servers in different Domino domains

When a sender mails a message, the Notes workstation looks for the recipient's mail address in the Domino Directory. If the recipient is a group and the group is listed in the Domino Directory, the Mail Router translates the group into a list of individual recipient names. The workstation appends each recipient's address to the message. Then the Mail Router on the server examines the document to find the location of the recipient's mail file, which is located on each recipient's home server. Based on the location of the recipient's home address, the Mail Router performs one of the following actions:

- ▶ If the recipient's mail file is on the same server as the sender's mail file, the Mail Router immediately delivers the message to the recipient's mail file.
- ▶ If the recipient's mail file is on a server that is in the same Domino named network, the Mail Router transfers the document to mail.box on the recipient's home server. Then, the Mail Router on the recipient's home server delivers the document to the recipient's mail file.
- ▶ If the recipient's mail file is on a server that is in a different Domino named network and if the sender's server has a Connection document that specifies how and when to connect to that server, the Mail Router uses the information in the Connection document to connect to the recipient's home server. The Mail Router transfers the message to the mailbox (mail.box) on the recipient's home server. Then, the Mail Router on the recipient's home server delivers the message to the recipient's mail file.
- ▶ If there are a number of servers between the sender's server and the recipient's server and the sender's server cannot connect directly to the recipient's server, the Mail Router temporarily stores the message in each server's mail.box until the server's Mail Router moves the message to the next server. This process repeats at each hop until the document reaches its final destination.
- ▶ If there is more than one possible route to the recipient's server, the Mail Router on the sender's server computes the least-cost route to the recipient's server based on the connection costs defined by Connection documents. Each server at each hop along the route computes the least-cost route to the destination.

To prevent routing loops, Domino sets a default maximum hop count of 25 for each message. That is, a mail message can make up to 25 server stops before the Mail Router returns the message to the sender. Each time the message passes through a server, the hop count decreases until the count reaches zero.

Potential performance impact

Mail routing depends on the mail infrastructure, the destination of a message, and the priority. To plan for mail routing, you must consider the following concepts:

- ▶ A domain groups Domino servers, which allows you to administer and control them easily. All servers in a domain have the same Domino Directory, which is the domain control center. The control center contains all of the documents necessary to route mail, connect servers, administer users, and configure servers. To send mail to another user in the *same* domain, enter the person's name in the To field of a memo. To send mail to a user in another domain, enter the person's name and the domain in the To field.
- ▶ A Domino named network is a subgroup of Domino servers that are physically connected. Domino treats a Domino named network as physically separate and distinct, even if it is connected to other servers and other Domino named networks. Mail routes automatically in a Domino named network and does not require Connection documents for delivery. A server Domino named network is listed in its Server record in the Domino Directory.
- ▶ A Connection document specifies how and when two servers connect, usually to exchange mail and to update common databases through replication. You need a Connection document from each server to the other to route mail.
- ▶ A routing table is a list of connections from a Domino server to all other servers it can contact. Domino assembles this table when the server starts by searching the Server, Connection, and Domain documents in the Domino Directory. Domino uses the routing table to determine the best, least-cost path to deliver mail.
- ▶ A mail.box file is the Domino mailbox. All outgoing and incoming mail on a server is held in mail.box until it is delivered. Mail is transferred from the sender's mail file to mail.box before it is routed. Routing moves the message from mail.box on the sender's server to mail.box on the recipient's server. Domino then moves the message from mail.box to the recipient's mail file.
- ▶ A home server for a given user is the server on which that user's mail file is located.
- ▶ A Message Transfer Agent (MTA) routes and converts messages from different formats, such as Multipurpose Internet Mail Extensions (MIME) and Notes mail.

Mail routing occurs on the backbone of your mail infrastructure, including your domain and Domino named network setup. When planning mail routing, consider these tips:

- ▶ Designate one server in each domain to connect to other domains.
- ▶ Designate one server in each Domino named network to connect to other Domino named networks.
- ▶ Use shared mail to reduce network traffic and the space needed to store mail.
- ▶ Use the mail trace feature in Domino to see the fastest route for mail and to debug routing problems.
- ▶ Schedule mail routing and replication together as tasks in Connection documents. This minimizes the number of Connection documents you need to create and reduces network traffic.

Suggested performance tuning

Because mail routing can involve the movement of large amounts of data, the way that you set up and schedule mail routing has significant performance consequences. A well-planned mail routing schedule that is tailored to your company's infrastructure and usage patterns prevents uneven server workloads and excessive network congestion.

Follow these guidelines when you set up basic mail routing:

- ▶ For mail routing to work properly, it is critical to set up replication so that the Domino Directory replicates to all servers in the Domino domain. Doing this ensures that the server, connection, and domain documents are up-to-date and consistent for all servers.
- ▶ Assign roles to restrict who can create Connection documents in the Domino Directory.
- ▶ You do not need Connection documents to route mail between servers in the same Domino named network because mail routing between them occurs automatically.
- ▶ You need two Connection documents, one in each direction (for example, a Server A connection to Server B and a Server B connection to Server A) to route mail back and forth between servers in different Domino named networks or in different Domino domains. To establish a connection between different domains, you need one Connection document in each Domino Directory.
- ▶ To minimize the number of Connection documents, designate only two or three servers in each Domino named network to route mail to other Domino named networks or to outside organizations.
- ▶ If you enable mail routing and replication on the same Connection document, you may want to create two separate Connection documents to schedule separate repeat intervals for mail routing and replication. This way, you can control the repeat interval for replication and mail routing separately and increase or decrease it, as needed.
- ▶ To locate and debug mail routing problems, use the Mail Trace feature.
- ▶ You can test mail routing after you create the Connection documents for each server. If the network is up and modems are available, you are ready to begin testing mail routing.

Increasing the number of threads for the Router task can improve mail-routing performance, especially on hub servers. When you send the following command, note the result:

```
COMMAND SENT: show stat mail.waiting  
MAIL.Waiting = 0
```

With Domino R5, you can create multiple mail boxes (MAIL.BOX files) to run several mail router tasks in parallel. A general recommendation is to start with two mail boxes and add more as you need them. The maximum number recommended to date is 10. To determine if you have message contention, issue the server console command `SHOW STAT MAIL.WAITING`. This gives you information about how much mail is waiting to be routed over a period of time. If you see that number beginning to grow, then you know that one of the paths you might need to pursue is to add another mail box to your server.

Figure 8-6 shows an example of the notes.ini server configuration document - Mail Routing.

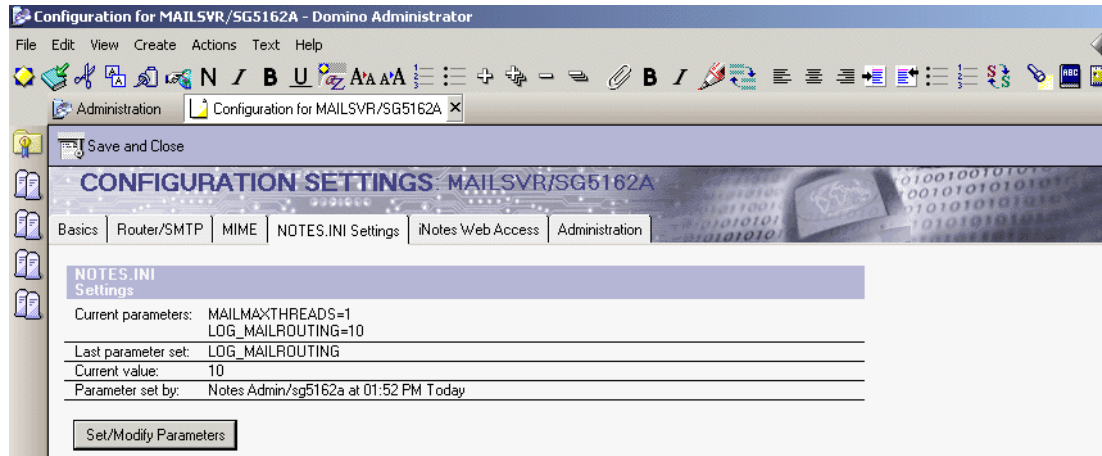


Figure 8-6 Notes.ini server configuration document: Mail routing

8.2.5 SERVER: The main server

The first job to be started after QNNINSTS is always a job called SERVER. It is the controlling job for all that transpires in the Domino server. It is the gate keeper for the open sessions.

In addition, the server job controls access to NSF (databases). It effectively provides a conduit for the Domino environment to function with all of the various jobs in the subsystem. In addition, the server job provides communication between any partitioned servers throughout the organization.

IOCP: A new way to use with threads

Like some of the other Domino tasks, the SERVER runs as a multi threaded job in OS/400. In the early releases of Domino for AS/400 (Domino for AS/400 5.0.1 and earlier), each session between a Notes client and the server was handled by one active thread of the SERVER job.

With the Quarterly Maintenance Update (QMU) 5.0.1.02, a substantial enhancement to Domino server performance was implemented by using asynchronous *I/O completion ports (IOCP)*.

The new server design enhancement allows Domino for iSeries to use fewer threads to support the users in the SERVER job. Instead of 1 thread per user, a small group of threads handles the work for all the users. We tested up to 10,000 users in a single partition, which required only 200 threads to be active. For more information on IOCP, see: <http://www-1.ibm.com/servers/eserver/iseries/domino/iocp.htm>

Threads are also further discussed in 4.8, “A brief discussion of threads” on page 62.

Suggested performance tuning

The performance suggestions for this job are further outlined in 6.1, “Tuning the iSeries server” on page 162. During the implementation phase of Domino, your organization needs to analyze functions that are going to be prioritized throughout the enterprise. It may be a good idea to partition a server for a specific purpose. If you do this, the information in Chapter 6, “Tuning the iSeries server for Lotus Domino” on page 161, can assist you in managing that individual subsystem from an iSeries perspective.

8.2.6 STATLOG: Database activity logger

The Statlog task runs on the server by default once a day at 5 a.m., unless you change the `ServerTasksAt5=` setting in `notes.ini`. It reports database activity for databases on the server in Database Activity Log entries in the Database - Usage and Database - Sizes views of the log file (`log.nsf`) and to the User Activity dialog box of individual databases.

The User Activity dialog box associated with databases summarizes database usage. By default, the Statlog server task activates this activity recording for server databases.

Potential performance impact

Reporting database activity in the User Activity dialog box of a database adds 64 K to the size of each database.

Suggested performance tuning

To conserve disk space and rely solely on the log file for database usage information, rather than viewing this information in User Activity dialog boxes for individual databases, edit the `notes.ini` file to include this setting:

```
No_Force_Activity_Logging=1
```

This prevents user activity from being automatically recorded in the User Activity dialog for all databases on a server. However, database managers can still enable user activity recording for individual databases.

8.2.7 UPDATE: The Indexer task

The UPDATE server task is included by default in the `notes.ini` setting `ServerTasks`. Therefore, it is loaded automatically at server startup. The UPDATE task updates all active views in a database when a user or a server task, such as the Replicator, updates any documents in the database. The UPDATE task also updates the full text index if one was created for the database.

Note: When you use the **show tasks** (`sh ta`) Domino console command, the UPDATE task appears as *Indexer*. However, the OS/400 job name and the entry in the `ServerTasks` statement in `notes.ini` are called UPDATE.

To save space on a server, UPDATE discards unused view indexes. A *view index* (not related to a full text index) is an internal filing system that allows Domino to create the most current list for a view. If a database designer has not chosen a discard index option for a view (by default, UPDATE discards a view that has been unused for 45 days), you can use the `notes.ini` setting `Default_Index_Lifetime_Days` to change when UPDATE discards unused view indexes.

Potential performance impact

You can run multiple UPDATE tasks using the `notes.ini` setting `Updaters` or on demand at the server console. A Domino administrator can start additional UPDATE tasks manually by entering the **load update** command at the Domino console. Or you can increase the number permanently by adding an `Updaters=x` statement to the `notes.ini` setting in a Server configuration document or adding it directly to the `notes.ini` file.

Note, however, that running multiple UPDATE tasks can severely impact server performance, so you should do this only if your server has adequate CPU power.

To understand the impact of the UPDATE task or tasks on the entire CPU utilization of iSeries server, you may log the platform dependent statistics in the statrep.nsf database. See Figure 5-26 on page 155 for an example and 5.3, “Combining performance data from Domino and iSeries” on page 152, for information on how to set it up. The statrep.nsf file can store the CPU utilization of up to four instances of UPDATE.

Suggested performance tuning

UPDATE can be CPU intensive. It is best to prioritize those objects that require updating. In addition, determine the type of updating required.

For detailed information about the update task, see 7.4, “Domino Database indexing: Controlling the UPDATE task” on page 185.

Note: Please see the Administration Help Guide, helpadmn.nsf, on your server.

Figure 8-7 shows an example of the notes.ini server configuration document - Update.

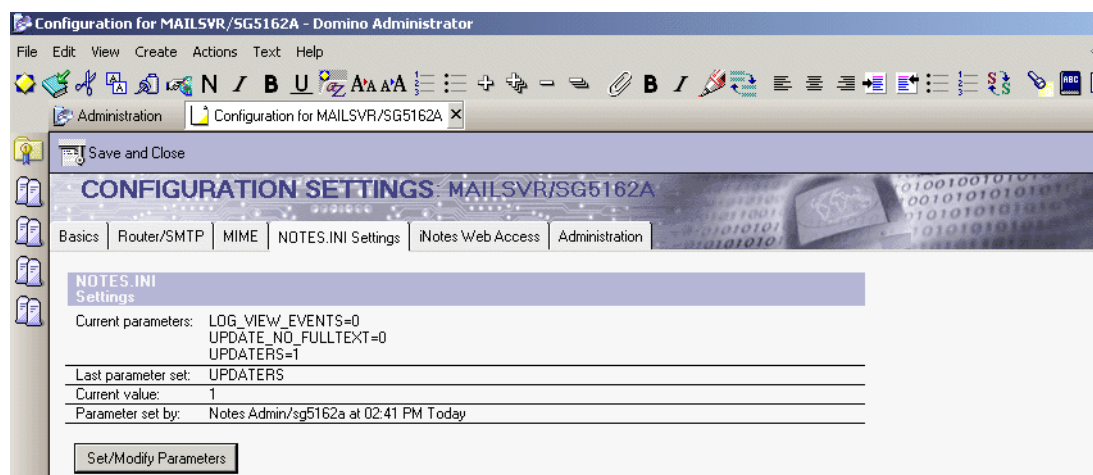


Figure 8-7 Notes.ini server configuration document: Update

8.3 Domino server jobs (often used)

This section covers advanced jobs that may not be currently used in your environment. However, exploiting the function sets that are running under these jobs will directly and positively impact the performance of your entire Domino environment.

8.3.1 CLADMIN: Cluster administration process

The Cluster Administration Process (CLADMIN) is responsible for the correct operation of all cluster components. When a server is added to or removed from a cluster, changes made to the Public Address Book are detected by the server, which then starts the Cluster Administration Process.

Potential performance impact

See 10.2.3, “Domino clustering” on page 283.

Suggested performance tuning

See 10.2.3, “Domino clustering” on page 283.

8.3.2 CLDBDIR: Cluster Database Directory Manager

The Cluster Database Directory Manager (CLDBDIR) task keeps the Cluster Database Directory up to date with the most current database information. Each time you add a database to a server, the Cluster Database Directory Manager creates a document that profiles the database and adds it to the Cluster Database Directory.

This database document contains information such as the database name, server, path name, replica ID, and other replication and access attributes. The information in this document is used by the cluster to determine failover paths and to control access to a database. It also manages databases with cluster-specific attributes such as databases marked out of service or pending delete.

Potential performance impact

See 10.2.3, “Domino clustering” on page 283.

Suggested performance tuning

See 10.2.3, “Domino clustering” on page 283.

8.3.3 CLREPL: Cluster Replicator

The Cluster Replicator (CLREPL) task is responsible for the tight synchronization of data among databases and their replicas in a cluster. Whenever a change occurs to a local database, the Cluster Replicator pushes the change to the other replicas in the cluster immediately, where the normal replicator starts working based on a schedule, for example every 60 minutes. Each server in a cluster runs one Cluster Replicator by default.

Domino administrators may increase the number of tasks on a server. To understand the impact of the CLREPL tasks on the entire CPU utilization of an iSeries server, you may log the platform dependent statistics in the statrep.nsf database. See Figure 5-26 on page 155 for an example and 5.3, “Combining performance data from Domino and iSeries” on page 152, for information on how to set it up. The statrep.nsf file can store the CPU utilization of up to four instances of CLREPL.

Potential performance impact

See 10.2.3, “Domino clustering” on page 283.

Suggested performance tuning

See 10.2.3, “Domino clustering” on page 283.

8.3.4 COLLECT: Statistics collector

The Collector task is useful for collecting statistics for one or multiple servers. The Collector task can be started on only one server, and that server collects statistics for that server and other servers that you designate. The collection configuration is done in the Statistics and Events database (events4.nsf), which is described in 8.3.5, “EVENT” on page 247.

For more information about the Collect task, see 5.2.2, “The Statistics and Reporting database (statrep.nsf)” on page 149.

Potential performance impact

The determining factor regarding performance depends on three factors:

- ▶ How often you want to collect statistics
- ▶ How often you want them analyzed
- ▶ Where to report them

Suggested performance tuning

To optimize performance, use the Collector server job only at one of the servers, to collect statistics of all servers because then it impacts only one server in the environment. This frees the other servers in your environment for end-user access.

Do not collect statistics more times per hour than you really need for analysis.

Figure 8-8 shows an example of Centralized Statistical Reporting.

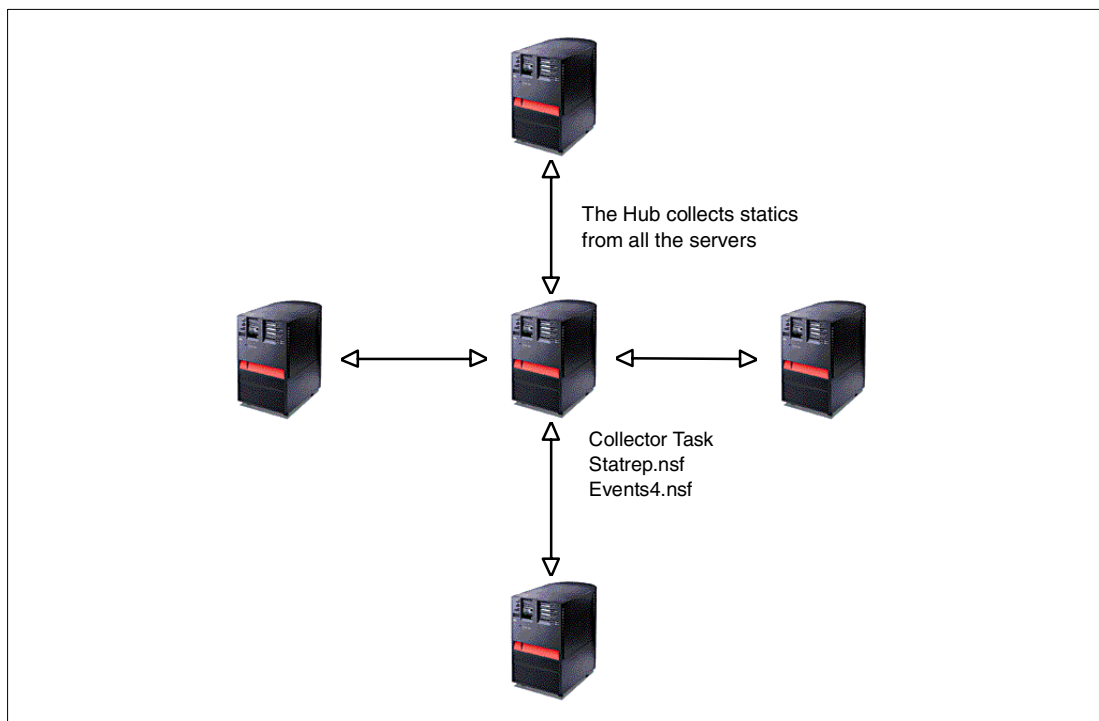


Figure 8-8 Centralized Statistical Reporting (Collector)

8.3.5 EVENT

The Event server task runs by default. You may also choose to run it manually at the server console if it is not already running. When the Event task runs, it creates the Statistics & Events database (EVENTS4.NSF) if this database does not exist. The Event manager was designed to provide the equivalent to an alarm for given thresholds and specific activity such as security violations.

The Event server task collects events and reports them to a person, database, or server-management program. You can monitor the following types of events:

- ▶ Communications events
- ▶ Mail events
- ▶ Miscellaneous events

- ▶ Replication events
- ▶ Resource events that relate to system resources
- ▶ Security events
- ▶ Server events that relate to server conditions such as tasks not executed or problems with the Server document in the Domino Directory
- ▶ Statistics including statistic alarms set by you and generated by the Report server task
- ▶ Update events (related to Indexer)

To configure which events to monitor and where to report them, you create Event monitor documents in the Statistics & Events database (eventse.nsf) for each type of event that you want to monitor. The Event task creates this database if it does not already exist.

In an Event Monitor document, you indicate:

- ▶ What type of event to report: Communications, mail, miscellaneous, replication, resource, security, server, statistics, or update
- ▶ Which events within this selected type to monitor based on severity level: Fatal, Failure, Warning (high), Warning (low), Normal, or All Severities
- ▶ Which servers to monitor
- ▶ Where to report monitored events: Person, database, server-management program, or OEM viewer

You can also use event monitoring to monitor access-control list changes and replication for specific databases on a server. To do this, you create ACL Monitor documents and replication monitor documents in the Statistics & Events database.

Potential performance impact

Typically, the event monitor does not significantly impact your Domino server performance. However, with a couple of thousand events to monitor, in addition to the ability to create events, which may be necessary for your organization, this overhead can begin to degrade Domino server performance.

Suggested performance tuning

Due to the overwhelming number of events that Event manager can monitor, you need a clear understanding during the architecture phase of your Domino server implementation. You also need a clear understanding of just what types of events are required to be monitored. In addition, due to the flexibility of the Event manager, you can turn the Event manager off and on when required. This may be the case in troubleshooting an intermittent issue or problem. In addition, you can partition an Administration server that can monitor (Events) on other servers, and eliminate any event resources from your production server (partition).

To remove events, enter the following command:

```
TELL EVENT QUIT
```

Then remove the Event task from the ServerTasks= parameter in the notes.ini file.

8.3.6 HTTP: The Web server

HTTP enables a Domino server to act as a Web server so browser clients can access databases on the server. The Domino server includes server technology that transforms Lotus Domino into a Web applications server. Domino combines the open networking environment of Internet standards and protocols with the powerful application development facilities of Notes. This enables you to develop a broad range of business applications for the Internet and intranet.

Potential performance impact

See 11.2, “HTTP performance tuning” on page 302.

Suggested performance tuning

See 11.2, “HTTP performance tuning” on page 302.

8.3.7 LOGASIO: Log asynchronous I/O process

The LOGASIO task is used to read and write transactional logs. It should never be loaded manually (it loads when transactional logging is enabled). But seeing a few threads of this task in an NSD is perfectly normal. It serves several purposes:

- ▶ It is a single process to do all log I/O, therefore requiring only that process to have file handles open on the transactional log.
- ▶ A separate process to allow the asynchronous writing of log records, therefore freeing up the requesting process unless it needs to know when the write is complete.
- ▶ Makes the logger appear as a shared resource accessed via shared memory and semaphores.

The LOGASIO process is used for writing while the server is running, and is used for reading and writing during recovery. The main point to understand is that LOGASIO is a service, not a module by itself. The main code in Notes signals LOGASIO to do writes or reads and that code affects databases.

Potential performance impact

See Chapter 13, “Domino transaction logging on iSeries” on page 371.

Suggested performance tuning

See Chapter 13, “Domino transaction logging on iSeries” on page 371.

8.3.8 REPLICA: The replication task

Replication is a server task that occurs between two servers in one direction. One server at a time calls a second server and checks for changes in common databases. The server then either sends changes to the other server or receives changes from it.

There are three types of replication:

- ▶ **Push replication:** Sends changes from the first server to the second.
- ▶ **Pull replication:** Receives changes from the second server to the first.
- ▶ **Push-pull replication:** Sends changes from the first server to the second, and then pulls updates from the second server to the first.

Controlling replication is a vital challenge. For example, a corporate policies database is controlled from a central office, with replica copies in all field offices. The company wants to restrict access to and replication of the database so that changes can only be made in the replica in the central office. The firm accomplishes this with Push replication so that the field office replicas receive changes from the central office replica, but cannot send changes to it.

Or, a company has replicas of a database both inside and outside its firewall, such as a customer comment database accessible through the Web. The company wants to prevent internal information and status tracking from replicating to the outside database. Therefore, it uses Pull replication to bring information from the database outside the firewall into the company.

Potential performance impact

Replication is a task that requires significant CPU and network resources. Plan your replication topology carefully to eliminate unnecessary connections, and make the most of each replication while keeping databases as up to date as possible. Replication depends on your network and server topology and varies greatly with organization size. For small businesses, replication involves only a few servers. Reducing the number of connections is less important due to the small number of total connections. For mid-size and enterprise organizations dealing with many servers, multiple sites, modems, and LAN and WAN connections, planning replication carefully is critical.

Domino replication is superior to other means of synchronizing databases in that you can control globally which replicas can make changes that propagate to other replicas. In addition, Domino offers a variety of replication strategies instead of limiting you to a superior-subordinate replication mode. Domino field-level replication greatly reduces replication time and the network and computer resources needed to accomplish this task. Domino is unique in that the network impact on replication can be eliminated by replicating between two servers under the covers of the iSeries server.

Scheduled process control by using Connection documents is a poorly designed feature and has a large negative performance impact on your server.

Suggested performance tuning

Here are some high-level overview items when managing the REPLICA task on the Domino environment:

- ▶ Be careful about over scheduling the server. When the server becomes overloaded, calls back up, mail is not delivered, and users may receive poor service from the server. Replication queue length has a maximum of six jobs queued.
- ▶ Check the log file (LOG.NSF) to see that databases are replicating properly, mail is routing promptly, and that the server is not overloaded. If necessary, modify the Connection documents and make adjustments until the problem is solved.
- ▶ If replications take too long, change the schedule to replicate more often so that there are fewer updates per replication.
- ▶ Schedule replication at off-peak hours. If you are replicating internationally, consider the time zones for the source and destination servers and plan accordingly.
- ▶ For the most dramatic improvement in performance, set up multiple replications so that a server, such as the hub, can replicate with multiple servers simultaneously. This should improve replication performance immediately. In this scenario, you would dedicate a server for replication in a spoke-hub typology. The users will realize greater performance since replication is no longer running on their spoke server.

- Use replication options to shorten replication times. For example, if a hub server replicates to 50 servers and you want to make sure an entire replication cycle occurs twice a day, limit the time that the hub connects to each server. Be sure to check the log to see which databases replicate completely and which do not.
- Set the replication priority to high, medium, or low to replicate databases of different priorities at different times. Set up replication groups based on replication priority.

Attention: This is very important! This allows for application categorization to prioritize replication during normal business hours.

- Do not schedule replication to occur at the same time for all servers! Staggered start times will increase performance.

Database managers assign a replication priority to databases to let Domino administrators schedule replication for databases based on their priority levels.

For example, you can schedule high-priority databases that are critical to business operations, such as Domino Directories, to replicate frequently. You can schedule low-priority databases to replicate during off hours.

To replicate databases by priority, edit the Replicate databases of the field in the Connection document. The default setting is Low & Medium & High. That is, Domino automatically replicates all databases that two servers have in common.

If two replicas are assigned different priorities, Domino uses the priority assigned to the replica on the server that initiates the replication. If a database is not replicating often enough, the database manager should increase its priority level. Complete the following steps:

1. Develop a policy that controls how database replicas are placed on the servers. Creating unnecessary replicas consumes system resources.
2. Check the Statistics & Events database for replication monitors that indicate server problems with replication.
3. Review the Replication log in LOG.NSF to ensure that replication errors are resolved since this will increase the speed of replication. Note the following command and result:

```
COMMAND SENT: sh stat replica.failed
Replica.Failed = 740
```

Figure 8-9 shows an example of the notes.ini server configuration document - Replicator.

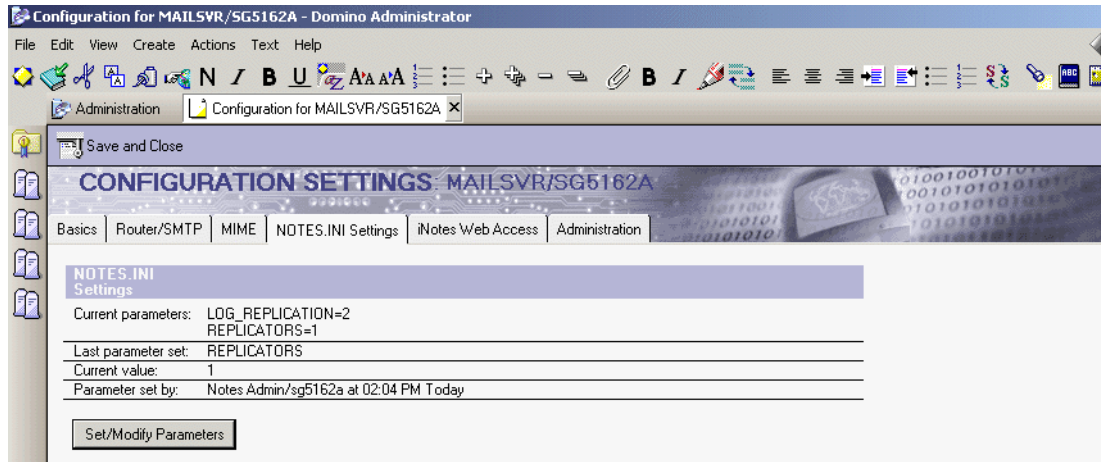


Figure 8-9 Notes.ini server configuration document - Replicator

8.3.9 SCHED: Schedule Manager

The Schedule Manager is responsible for returning meeting times and dates and available invitees to requests for free time searches.

The first time that the Schedule Manager task starts on a server, it creates a Free Time database (BUSYTIME.NSF) on that server. It also creates an entry in the Free Time database for each user whose MailServer field (in the Person document) is set to that server name. Schedule Manager runs continually. Each time users make a change to their calendars, it enters the change in the Free Time database. If you delete the Free Time database, next time when you start the server, the Free Time system recreates the database and enters information for all users whose mail files are on that server.

When a user invites people to a meeting and chooses to search for the invites' free time, Schedule Manager searches the Domino Directory for the location of each invited's mail server and mail file. It then searches for schedule information for each invited in the Free Time database on each invited's mail server. Schedule Manager returns a list of meeting dates, times, and a number of available invites. The list is in descending order from the date and time, with the most number of attendees to the least number of attendees available.

The Free Time system consists of two server tasks: Schedule Manager (SCHED) and Calendar Connector (CALCONN). The server installation program automatically adds both tasks to the server notes.ini file when you install Domino.

Potential performance impact

The SCHED command opens another database file into memory for every user session on that server. It is imperative that the number of file handles on your system can cope with the additional load.

Suggested performance tuning

If free-time lookups are not required, by your organizational requirements, then it can be removed. Enter the following command:

```
tell sched quit
```

Then remove the Sched task from the ServerTasks= parameter in the notes.ini file.

8.3.10 SMTP: Internet Mail Router

The Simple Mail Transport Program (SMTP) allows the Domino server to transfer SMTP messages between SMTP networks (both Internet and intranet) and Notes, X.400, and cc:Mail users. Other Internet mail clients that use Domino as a message store, such as POP3 and IMAP clients, also require SMTP support for transferring messages. SMTP allows the Domino server to transfer SMTP messages between SMTP networks (both Internet and intranet) and Notes, X.400 and cc:Mail users. Other Internet mail clients that use Domino as a message store, such as POP3 and IMAP clients, also require SMTP support for transferring messages.

To transfer mail reliably and efficiently, SMTP defines mechanisms for relaying mail between networks on the Internet. Previously, you would configure the SMTP MTA on a few dedicated Domino servers to act as gateways for your Internet mail, converting Notes mail to Internet mail, and vice versa. All Internet mail had to transfer through these servers.

With Domino Release 5, you no longer need to install a separate MTA. All Domino servers now have native SMTP capabilities, so all servers can transfer mail directly to and from the Internet. As mentioned previously, conversions to and from MIME are no longer necessary since the Release 5 client can send and receive native MIME messages. Of course, the server can still convert messages for pre-Release 5 clients. Domino also supports Extended SMTP (ESMTP) for delivery notifications for Internet messages, for example, reporting whether deliveries are successful, fail, or are delayed. In addition, Domino Release 5 allows you to secure SMTP connections using TCP/IP or a TCP/IP port secured with Secure Sockets Layer (SSL). You can require a name and password either unencrypted over TCP/IP or encrypted over SSL. This is part of the Domino stack and does not use the OS/400 SSL.

Potential performance impact

See Chapter 7, "Tuning Lotus Domino for better performance on iSeries" on page 183.

Suggested performance tuning

See Chapter 7, "Tuning Lotus Domino for better performance on iSeries" on page 183.

8.4 Domino server jobs (very likely not used)

Some of the Domino tasks being started automatically with your server may not be necessary in your environment. This section lists those Domino tasks that are seldom used, but very likely started, because they were needed in previous releases or because of a configuration setting not fully understood by the person who set up the server.

8.4.1 BILLING

The Billing server task enables a Domino server to track specific server activities. The Billing server task collects the billing information provided by the server and records the data for billing purposes.

Domino servers can track specific activities for billing. Each type of activity is designated by a billing class. You select which activities you want the server to track by adding classes to the BillingClass settings in the notes.ini file. The Domino server places this information in the billing message queue. The Billing server task then periodically polls the message queue and removes the billing records to a database or file that you specify.

You can use the Domino-supplied Billing task or you can design your own server task using the API. A Billing server can send additional data to the billing message queue using the BillingWrite API.

Potential performance impact

Billing requires large CPU utilization. If you must use billing, these are some of the parameters that you must use to control the impact on CPU utilization:

- ▶ **Billing classes** (Database Document, Mail, Replication, Session, Agent): Provide a way for the organization to gather specific billing data.
- ▶ **BillingAddInRunTime parameter**: Specifies the duration of the billing task.
- ▶ **Frequency of billing task start**: Specifies how frequent the session or database is stamped or recorded. The larger the number is, the better the server performance is. Given these factors, the less detail billing information is available.

Figure 8-7 shows an example of the notes.ini server configuration document - Billing.

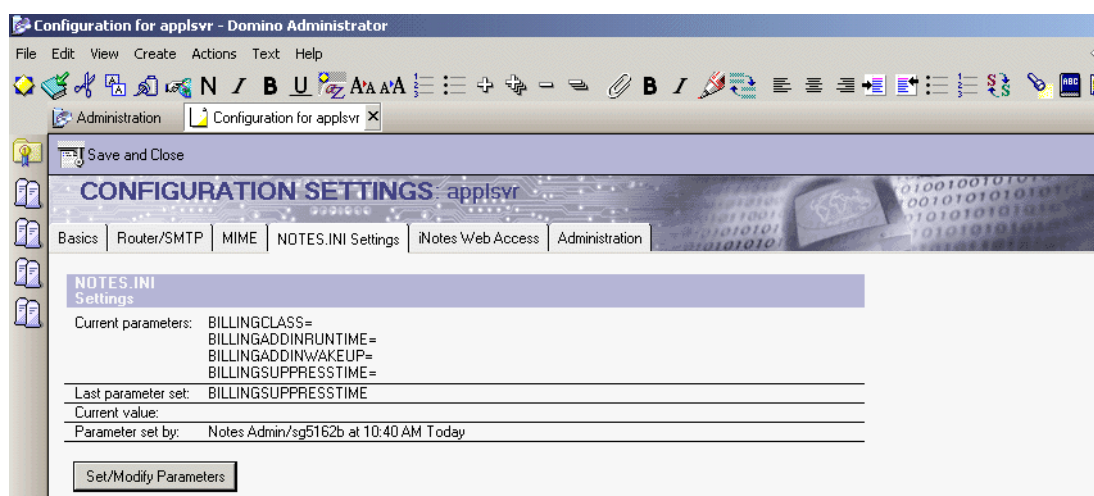


Figure 8-10 Notes.ini server configuration document - Billing

Suggested performance tuning

Note: Billing requires significant CPU resources. It is not advisable to turn on billing.

Perhaps an alternative is to create an agent (application) that interrogates the native Domino server logs that already are created by using the server. These logs could easily be replicated to a consolidated server, and the agent could be written to run at specific times during the day against the replicated logs (databases). This implementation would provide greater control of when the agent runs. In addition, it would not impact the production systems because billing is not running on those systems.

8.4.2 CALCONN: Calendar connector

A user on one server may request free-time information for a user whose calendar information resides on a different server. When this happens, the Calendar Connector task uses the Domino Directory to find the paths between the user's and the invitees' mail servers.

You must have adjacent domain documents to ensure real-time connectivity between domains for free-time queries. Domains that share scheduling information must have direct access to each other. Non-adjacent domains do not provide real-time connectivity. You do not need to create new adjacent domain documents if they already exist for the domains that will share calendar information.

The Free Time system consists of two server tasks: Schedule Manager (SCHED) and Calendar Connector (CALCONN). The server installation program automatically adds both tasks to the server notes.ini file when you install Domino.

Potential performance impact

Exposure to system resources for the Calendar Connector job is not great. However, if your organization requirements encompass many different users on different servers, you can realize resource consumption. Typically, this consumption does not significantly reduce overall system resources or response times.

To analyze the impact of calendar and scheduling on the Domino server, issue the following command at the Domino server console and note the results:

```
COMMAND SENT: sh stat calendar
Calendar.Total.All.Appts.Reservations = 1
Calendar.Total.All.Users.Resources = 4
Calendar.Total.Appts = 1
Calendar.Total.Reservations = 0
Calendar.Total.Resources = 0
Calendar.Total.Users = 4
```

Suggested performance tuning

If your organization has one home server for everyone, there are no other servers throughout the organization to do free-time lookups. Therefore, you can disable CALCONN.

If your organization is properly configured, chances are that most of the department individuals would be on one server. Assuming that these are the individuals that need to check free-time lookup, again CALCONN can be disabled.

However, if you have multiple home servers with many departments, individually assigned to different home servers, you have a couple of options:

- ▶ Determine when free-time lookups are to occur (based on requirements).
- ▶ Limit (turn off) CALCONN for free-time lookups across server boundaries.

After all this, if organization requirements must allow free-time lookups across server boundaries, CALCONN must be on.

To remove CALCONN, from the subsystem, type the following command:

```
TELL CALCONN QUIT
```

Then remove the CALCONN task from the ServerTasks= parameter in the notes.ini file.

8.4.3 QNNINADD: Directory synchronization add-in task

QNNINADD is the Domino server task that starts Directory synchronization services, which allows you to keep the iSeries server distribution directory synchronized with the Domino Directory (previously known as the Public Address Book or the Name and Address Book). This additional task job calls a specific configuration of Domino server. It is a very brief job that begins and ends during the start of the server. Its performance impact is very brief and without CPU performance degradation.

Directory synchronization was installed and started by default in earlier releases. If it is active at your system and you don't use Directory synchronization, there are steps you have to perform to remove the function.

For information about how to remove Directory synchronization, see 7.17, "Domino for iSeries specific notes.ini parameters" on page 224.

8.4.4 REPORT: Statistic reporter

The Report task no longer exists in Domino server Release 5. It has been replaced by the Collect task. For information about the Collect task, see 8.3.4, "COLLECT: Statistics collector" on page 246.

8.4.5 STATS: Statistics on demand via e-mail

The STATS task allows you to collect statistics on demand from a server and mail them to yourself or other people. Domino loads the STATS task by default on every server. However, to mail statistics, you must manually load the task on the server so that Domino creates a mail-in database that has the database name `servername Stats` and the file name `STATMAIL.NSF`.

The STATS server task was designed to collect statistics for Notes version 3.x. The Collector server task supersedes the STATS server task. The STATS server task, by default, starts automatically when a server is started. To prevent the STATS server task from starting automatically, remove the STATS task from the `ServerTasks=` parameter in the `notes.ini` file:

```
ServerTasks=Replica,Router,Update,AMgr,Adminp,Sched,Ca1Conn,Event,QNNINADD,Stats
```

Potential performance impact

STATS uses little system resources.

Suggested performance tuning

There are no suggestions available for performance tuning.

8.5 Domino server database housekeeping jobs

Domino server database housekeeping jobs cover additional jobs that are run in a batch environment. That is, they are instantiated typically by the Domino system administrator or an agent and run for a specified period of time.

8.5.1 COMPACT

The Compactor task compacts all databases on the server to remove white space and to free disk space. Deleting documents and attachments from a database leaves blocks of unused space within the database. As new documents are created, they fill in the unused space.

To compact databases, you can use the administration panel or load the Compactor task from the server console or a program document. When you load the Compactor task, you do not need designer access to databases that you compact, and you can use arguments to control how compacting works.

Potential performance impact

Compaction is a highly CPU-intensive task. In addition, it needs additional DASD due to the creation of a temporary copy of the database.

Suggested performance tuning

Because compacting large databases can take some time, be sure to run compacting during off-peak hours so that users do not experience performance problems. Compacting a database creates a temporary copy of it. Therefore, the server must have enough disk space available to store the copy during the process.

There is a notes.ini file setting that enables or disables compacting of the servers mail.box. Without setting the parameters on this setting, compaction is done routinely when the COMPACT server task runs. To manage server performance, you can flag the notes.ini file for:

MailCompactDisabled=value

Assign a value of 0 or 1. In this parameter, note these options:

- ▶ 0 = Enables compacting the mail.box
- ▶ 1 = Disables compacting the mail.box

For a detailed description of the COMPACT task, see “Compacting a Domino database” on page 385.

8.5.2 UPDALL

The UPDALL task updates all changed views or full-text indexes for all databases. The UPDALL server task updates all views that have been accessed at least once and all full text indexes for all databases on a server. UPDALL runs by default at 2:00 a.m., determined by the notes.ini setting ServerTasksAt2. If you run UPDALL manually or through a program document, you can include arguments that determine how the task works. For example, you can update views, but not full text indexes. This task also detects and rebuilds any corrupted indexes.

Potential performance impact

You can run multiple UPDATE tasks using the notes.ini setting Updaters or on demand at the server console. Running multiple UPDALL tasks can impact server performance. Therefore, you should do this only if your server has adequate CPU power.

Suggested performance tuning

UPDALL requires significant CPU resources. It is not advised to run this task during normal business hours. You can use the notes.ini setting Update_No_Fulltext=1 to prevent full text indexing on a server. In addition, you may want to develop a database design strategy that calls for an update to database views on a scheduled or triggered basis.

See 7.4, “Domino Database indexing: Controlling the UPDATE task” on page 185, for more information.

8.5.3 Fixup

The Fixup server task checks for and fixes corrupted databases. Although Domino automatically checks for and fixes a corrupted database, if necessary, each time a user opens a database, Fixup helps reduce the number of corrupted databases that users encounter.

In addition, manually loading Fixup from the server console or administration panel rebuilds any corrupted views or folders. Corrupted views and folders are not fixed when Fixup runs automatically at server startup or when users open corrupted databases.

Fixup cannot run on open databases and databases cannot be opened when Fixup is running on them. However, when Fixup encounters an open database (for example, log.nsf, mail.box, or names.nsf), Fixup reports this to the log file.

Potential performance impact

Fixup is a CPU-intensive task. Multiple Fixup tasks run simultaneously at server startup to reduce the time required to fix databases. The number of Fixup tasks that Domino runs by default at startup is equal to two times the number of processors available on the server. The default behavior should be adequate for most servers. To change the default, edit the notes.ini file to include the Fixup_Tasks setting. The actual number of tasks run is the smaller of the configured number of tasks than can run and the number of databases that require fixing. For example, if you set Fixup_Tasks to 4, but only one database requires fixing, then only one Fixup task runs.

Suggested performance tuning

Since Fixup uses significant CPU resources, run it manually only when necessary. It must be run outside of normal business hours whenever possible. In addition, this task can be executed by database name. If you have a specific database that is corrupted, run FIXUP against that database during normal business hours. The syntax for Fixup can be found in the Domino Administration Help database (ADMINHELP.NSF).

You may create a program document indicating to start Fixup during after hours on a specific day. If this program document were set up to run Sunday morning at 2:00 a.m., it would reduce the exposure of Fixup running against any open files, and therefore, produce a more complete task!

8.6 Series SMTP: AnyMail/400 Mail Server Framework (MSF)

iSeries SMTP used together with AnyMail/400 Mail Server Framework (MSF) job are described in “SMTP and AnyMail/400 Mail Server Framework (MSF) jobs” on page 427.

8.7 Domino server QNNxxxx programs and QSYSWRK

Domino server QNNxxxx and QSYSWRK are described in “Domino for iSeries server programs QNNDxxxx and QSYSWRK” on page 432.

Most of the QNNDxxxx programs handle the directory synchronization services. Directory synchronization allows you to keep the iSeries server distribution directory synchronized with the Domino Directory (also known as the Public Address Book or the Name and Address Book).

Directory synchronization was installed and started by default in earlier releases. If it is active in your system and you don't use Directory synchronization, there are steps you have to perform to remove the function.

For information about how to remove Directory synchronization, see 7.17, “Domino for iSeries specific notes.ini parameters” on page 224.



Integration with DB2 performance tips and techniques

To ensure the optimal performance of your Domino servers, it is essential to consider the impact of application design and function. Domino is an application environment that uses a client/server architecture to provide functions to users through a common interface. The interface can be either Web browsers (HTTP) or the Notes client (NRPC).

The nature of this client server architecture allows developers to design applications where the workload can be distributed to the client workstation or take advantage of workloads centrally on the server. Application design allows developers this flexibility for each design element of an application.

Taking the above point into consideration, it can be said that the design impact of applications running under Domino can greatly impact all facets of Domino server performance. Designing (or re-designing) the application to optimize performance can often greatly improve end-user performance, and at the same time, reduce overhead to the Domino server.

There are many different factors that can affect the perceived performance of a Domino application:

- ▶ Application design and implementation
- ▶ Interaction with external systems
- ▶ Domino server configuration
- ▶ Network configuration
- ▶ Traffic generated from other applications
- ▶ Server and client operating system configurations
- ▶ Server and client hardware configurations

Most commonly, the biggest factor involved in application performance involves how the application has been designed and implemented. If the core application has not been designed properly, the application will not scale with multiple user loads and therefore will not meet business requirements.

The redbook *Performance Considerations for Domino Applications*, SG24-5602, contains many details on designing a Domino application for optimal performance. It is an excellent reference when creating a Domino application. It contains information on design considerations, programming considerations, and performance aspects of different Domino design elements (views, agents, forms, ...). Using this redbook, you will gain a very good understanding of what aspects to consider and successfully implement to have a high performing Domino application.

However, that redbook does not cover the integration of Domino and DB2. Because this integration is so key for the iSeries server, we include the information in this chapter on designing a high performing Domino application that interacts with DB2.

You can find more in-depth information about the integration of Domino and other iSeries applications in:

- ▶ *Lotus Domino for AS/400: Integration with Enterprise Applications*, SG24-5345
- ▶ *Developing e-business Applications Using Lotus Domino for AS/400*, SG24-6052
- ▶ *New Enterprise Integration Functions for Lotus Domino for AS/400*, SG24-6203

9.1 Choosing a storage container

Domino and DB2 UDB for iSeries are very different data storage facilities. Because of these inherent differences, the performance and scalability of these two options are different. This section starts out by exploring the differences in how DB2 UDB and Domino are implemented and then moves into the details on how best to integrate them.

9.1.1 Domino as a storage container

Domino is a very flexible storage container. It is the most popular secure, distributed, collaborative application environment for document-oriented information. You can store both structured and unstructured data in Domino. Structured data includes text fields that always contain text data, numeric fields, date and time fields, or name fields.

Unstructured data can be stored in a rich text field. One document may contain a photo attachment in this field, the next document may contain simple text in the field, and the third document may contain some text in addition to a movie attachment.

The ability to combine both structured and unstructured data in Domino makes it a unique data storage option.

The actual physical structure of a Domino database is a flat file. On an iSeries server, those files are stored as stream files (OS/400 object type STMF) in the Integrated File System (IFS). They contain collections of documents that are linked by some common purpose. Each of these documents is composed of a set of fields that can contain either structured or unstructured data.

Another aspect of allowing unstructured data within a Domino database is the fact that not all documents need to contain the same fields. This is very different to an SQL table, where each row has to have the same layout. Each Domino document is self descriptive. That is, it not only contains the contents of the fields, but also each field's attributes.

Another key difference for Domino databases is that they contain the data and the design elements that are required to manipulate the data. A Domino database typically contains forms and views that let you add new documents into a Domino database and look at the existing documents in the database.

Forms and views also implement the main application logic in the form of simple actions, function language (@xxx) procedures, LotusScript, or Java code. In most cases, the logic is triggered directly by user activities, like displaying or entering data, but a Domino database can also contain agents that can be started by a time schedule or an event, for example, such as incoming mail.

Domino databases are best used when you need to manage documents and when replication of both the application and the data is required.

9.1.2 DB2 UDB as a data storage container

DB2 UDB is a very different storage container than Domino. It is a top-rated relational database management system (DBMS) that is designed to handle ad hoc structured queries and transaction processing required to manage and operate a business of any size. The focus of data stored in DB2 is point-in-time capture of structured data and basic business transactions.

The DB2 implementation on iSeries

The physical structure of DB2 databases is quite different from Domino. The relational database is represented internally on the iSeries server as a set of objects that are managed and optimized below the *Technology Independent Machine Interface* (TIMI; formerly called Machine Interface (MI)).

Every DB2 UDB database on the iSeries is comprised of a cursor and a data space. The data space contains the actual data and the cursor provides a pointer into the data residing in the data space. If an index for an SQL table (or physical file) has been created by a programmer, then a data space index is used internally to implement that index.

The data stored in DB2 is basically a named collection of records. Each record is a collection of fields that represent named items of data that represent attributes of an item. Another way DB2 UDB differs from Domino in data storage is that DB2 contains structured data.

The data in each record in DB2 contains the same type of data in each field. An integer field always contains integer data, a numeric field always contains numeric data, and so on. The only exception to this rule is the use of Binary Large Objects (BLOB) in DB2. These data types allow unstructured data to be stored in relational databases such as DB2.

SQL and iSeries terminology for DB2

When talking about DB2 UDB on iSeries, compared to other DB2 platforms, the iSeries has a very unique implementation in several aspects. One differentiator from other platforms is the choice of the programmer to access DB2 data through SQL or natively on a record-by-record base. The underlying internal implementation is the same in either case. However, the terminology is different.

Note: A Domino developer who is experienced with DB2 access on platforms other than OS/400 most likely does not know the terms physical and logical file, record format and field. Even the term *file* in the context of a relational database is misleading, since on most other platforms, this implies to be a flat, non-database file.

Native iSeries terminology talks about:

- ▶ Physical files
- ▶ Logical files

Physical files always have just one *record format*. That is, all records in the physical file have the same layout in terms of the number and data type of fields. For an SQL programmer, this is an *SQL table* that contains *columns* and *rows*.

This is quite different from Domino since Domino databases can have multiple record layouts in the same physical database. In fact, Domino does not have the concept of a record format in the sense that all documents could contain a different set of fields¹.

The value of a DB2 Index

Logical files provide an alternate way to access data in one or more physical DB2 UDB files (or SQL tables). They are commonly used to:

- ▶ Select a subset of the records in a physical file. In SQL terminology, this can be compared to an SQL view.
- ▶ Merge records from multiple physical file members.
- ▶ Join related records in two or more physical files.
- ▶ Provide an index so records can be retrieved in a particular order. In SQL terminology, this can be compared to an SQL index. This is a *huge* performance benefit!

When you create an OS/400 logical file and define one or more fields as a *key* or when you create an *SQL index*, the system internally builds and maintains a data space index to allow extremely fast access to data records and SQL rows through the usage of a binary radix tree.

Relational databases are best used when there is a need to access large quantities of data and performance is key. Relational databases scale well beyond Domino databases. One of the most common problems we have seen is the mistake of moving data from DB2 into Domino. This typically happens because the Domino application developers are not familiar with DB2 so they just copy the data into Domino where it is accessed.

In smaller databases, this is usually not a problem. But as the size of the database grows to thousands of records and hundreds of end users are trying to concurrently access this data, performance quickly becomes an issue. For this reason, it is very important to leave the data in DB2 where it belongs and use the best performing option for your environment to work with the data from the Domino application.

9.1.3 Why Domino does not scale like DB2 UDB

We have looked at both the physical structure of the Domino and DB2 storage containers. As you have seen, they are very different. It is because of these differences that Domino and DB2 UDB scale differently.

The main difference is in the use of keys. DB2 stores data in tables and can use many keys (or indexes) to quickly build dynamic views of related data. A relational database can display views that dynamically combine data from several different tables, relying on multiple keys to relate the data in one table to the data in another table. This dynamic compilation of data happens very quickly.

¹ This might sound strange assuming all documents were created through the same format, but fields may be added or deleted later or entire documents from a different database may be pasted or copied into the database containing fields, which are not defined in any form of this database.

Domino, on the other hand, stores data in documents, and a Notes document has only one key, the document unique ID (UNID). In Domino, you cannot designate one or more fields as a *key*. Domino creates and maintains indexes only based on the first sorted field within a Domino view. This makes it impractical for a developer to code the types of high performing dynamic looks that can be done in DB2. See 7.4, “Domino Database indexing: Controlling the UPDATE task” on page 185, for a more in-depth discussion on the impact of Domino indexes on response times.

This inherent difference between how keys can and cannot be used makes the scalability of Domino and DB2 UDB divergent.

9.2 Integrating Domino and DB2 UDB

In today’s business environment, most organizations face the challenge of having data and information stored in a number of different application systems for Enterprise Resource Planning (ERP), Customer Relationship Management (CRM), Supply Chain Management (SCM), and others. There is a need to access data from all of these systems and build business applications based on that data.

Luckily Domino provides many different options for connecting Domino to enterprise data and information. The benefits of this integration are huge, allowing for rapid connectivity to all of the most popular RDBMS, transaction processing monitor systems, or directly to some of the most popular ERP, CRM, or SCM solutions, such as SAP R/3, Lawson, Infinium, and others. Additional benefits include:

- ▶ Access to relational data without having to replicate it to a local site
- ▶ Existing applications do not need to be rewritten to use an alternate data source
- ▶ Web browser access doesn’t require an ODBC connection on each client and Web agents run on the server for faster data access

The options for integration include using a choice of programmatic or declarative (forms-based, visual data mapping) methods. The following sections compare the various integration options and provide guidance on when each method is most appropriate.

9.2.1 Integration options

As already mentioned, there are many different ways that Domino can integrate with DB2. The methods are described in the following sections and include:

- ▶ “Domino Enterprise Connection Services (DECS)” on page 264
- ▶ “Lotus Enterprise Integrator (LEI)” on page 264
- ▶ “@DB commands” on page 269
- ▶ “LotusScript Data Object (LS:DO)” on page 270
- ▶ “Lotus Connector LotusScript Extension (LC LSX)” on page 270
- ▶ “Java Database Connectivity (JDBC)” on page 270

Lotus Enterprise Integrator (LEI) is a separately charged product. All other functions listed above are part of the Domino server or Notes client.

As we move down the list of integration options, we move from graphical, non-programmatic methods to programmatic options. In addition, as we move down the list of techniques for accessing data, complexity grows, but so does performance and scalability. We look at each of these integration options in more detail and provide performance advice on when each of these options is most appropriate.

Domino Enterprise Connection Services (DECS)

Domino Enterprise Connection Services is a graphical tool based on a Notes client that is very easy to use. It provides access to relational data with no programming required and is a fairly simple solution, offering a one-to-one relationship between Domino and DB2. DECS provides real-time access to data, leaving the relational data in DB2 UDB.

Using graphical wizards, the fields on a Domino form are mapped to fields in one or more relational tables. When a Notes or Web user is working with data through the DECS-enabled form, the data is retrieved, updated, inserted, or deleted real-time in DB2 UDB. The only data stored in the Domino database is the key data that allows for unique access to the data stored in DB2.

Advantages and disadvantages of using DECS

DECS is great for single record lookup. It does not work for retrieving multiple records and then manipulating this data in a Domino application.

Performance may suffer when using DECS in these scenarios:

- ▶ Many fields on a form are populated from DB2
- ▶ Hundreds of connections are open from Domino to DB2

DECS has been successfully implemented in many customer environments and provides quite good scalability even for hundreds of users. However, when you get into the situation where you are populating many fields on a Domino form through DECS or when there are hundreds of DB2 UDB database connections open through DECS, the performance and throughput of this option breaks down quite quickly.

Lotus Enterprise Integrator (LEI)

Lotus Enterprise Integrator, formerly known as NotesPump, is another graphical tool that makes integrating with DB2 easier and also requires no programming. Unlike the other functions described here, it is a product that needs to be purchased separately from Domino.

LEI is best utilized when the need is to download massive quantities of data from relational tables in batch mode (as opposed to DECS, which always needs the user to access the data). It is most commonly used to replicate data between Domino and DB2, allowing Domino users to replicate data from the Domino application to their laptop so they can work in a disconnected environment. As such, LEI is a great option when access to relational data is needed when users must work offline.

LEI can be used very successfully to replicate or synchronize data between Domino and DB2 when the replication activities are run during off-shift hours or the replication involves smaller quantities of data. Performance becomes a concern with LEI when:

- ▶ The number of records to download or synchronize is greater than 10,000.
- ▶ There is a need to run replication or direct transfer activities during “prime shift”.
- ▶ Sort sequence is different between Domino and DB2 for replication activities.
- ▶ Many replication activities are running concurrently.

The following paragraphs explain some techniques to improve LEI performance.

Increased run time because of different sort sequence for ASCII and EBCDIC

When using a LEI Replication Activity to replicate DB2 data from an iSeries server to a Domino database, using a mixed alpha-numeric key, that is alpha characters *and* numeric digits at the same position, you find that LEI performs unnecessary record inserts and deletes. What causes this behavior?

This is caused by the different sort sequences used by EBCDIC-based systems and Domino servers. Specifically, EBCDIC-based systems sort mixed alpha-numeric data by alpha characters first and then by numeric characters. Domino, on the other hand, sorts mixed alpha-numeric data by numeric characters first and then by alpha characters. When this occurs, LEI inserts records that do not match the sorted order that is presented by the DB2 connector and then removes the same records after Notes presents the same records.

This is a fairly common occurrence and can potentially cause an increase in processing time for the Activity or the insertion of duplicate records, if deletions are skipped on the Notes connection.

The purpose of this section is explain how the iSeries server sorts data and how you can make adjustments to the sorting sequence. Domino for iSeries, although running natively on OS/400, still sorts data as an ASCII-based system or, to be more exact, based on the Lotus Multi-Byte Character Set (LMBCS) a superset of ASCII.

With this information, you can understand how to change the sort order of data when retrieved from DB2 UDB for iSeries. Ideally, this causes the data that is retrieved from DB2 to be presorted before it is presented to the LEI Activity. This eliminates the need for using an *Order MetaConnector* against a DB2 connector when retrieving DB2 data from an iSeries server.

By default, the iSeries server is not configured to sort data, which causes an order mismatch when retrieving data from DB2 tables via Lotus Domino, especially with LEI or DECS. This is most noticeable when running an LEI Replication Activity from DB2 to Notes, where unnecessary record insert and deletes may occur. LEI provides an Order MetaConnector that can be used to address this specific issue. However, the use of this connector affects the performance of LEI, and, if the result set is too large, can cause the Activity to fail.

By changing the QNOTES user profile, and configuring it to use a specific sort sequence, you can avoid the issue entirely. When the data is retrieved, it will be presorted before it is presented to LEI. Therefore, because the data is presorted, there is no need to use an Order MetaConnector against a DB2 connection that references DB2 for iSeries. This is important because it eliminates the need for the connector against the DB2 connection. It also prevents the unnecessary insertions or deletions of records during a Replication Activity, therefore, enhancing overall LEI performance.

Important: This has no effect when non-alpha-numeric data is involved, such as #, @, \$, %, &, etc. Also, language-specific special characters (ä, ü, ö, ß, ... , é, è, ..., ø, ... etc.) are *not* considered to be non-alpha-numeric characters. If non-alpha-numeric data is involved, then an Order MetaConnector against the Notes connection, specifying a character set of ASCII, needs to be used.

The sort sequence (SRTSEQ), along with the language ID (LANGID) parameter in the Create User Profile (CRTUSRPRF) and Change User Profile (CHGUSRPRF) commands, is used to determine the sort sequence table to be used for sorting character data. You can change SRTSEQ to any one of the values listed in Table 9-1.

Table 9-1 Values for the sort sequence parameter of an OS/400 user profile

Value	Meaning
*HEX	No sort sequence table is used. The hexadecimal values of the graphic characters are used to determine the sort sequence (a binary sort). This is the only sort sequence available for Double Byte Character Set (DBCS) data.
*LANGIDSHR	The sort sequence table may use the same weight for multiple graphic characters. The shared-weight sort table associated with the language specified in the LANGID parameter is used. This sort applies only to SBCS data.
*LANGIDUNQ	The sort sequence table contains uniquely weighted graphic characters. The unique-weight sort table associated with the language specified in the LANGID parameter is used. This sort applies only to SBCS data.
Qualified sort sequence table name	The name and library of the sort sequence table to be used. This value allows you to specify a sort sequence table other than those associated with the language specified in the LANGID parameter. This sort sequence table applies only to SBCS data.

With the above settings in mind, you can make the comparison shown in Table 9-2 between the iSeries and Order MetaConnector settings.

Table 9-2 OS/400 user profile and LEI Order MetaConnector comparison

OS/400 user profile (SRTSEQ)	LEI Order MetaConnector
*HEX	Binary
*LANGIDUNQ	Sort with case sensitivity
*LANGIDSHR	Sort without case sensitivity

Sort sequences are supported only for SBCS* EBCDIC languages. The example in Table 9-3 shows the relative sort sequence weights for characters sorted using the default sort sequence tables shipped with the system. When looking at these tables, consider the following points:

- ▶ Several tables shipped with the system represent a single sort sequence, each encoded with a different coded character set identifier (CCSID) value. Not all of the characters in a given sort sequence exist in every CCSID in which the sort sequence is encoded.
- ▶ Use the language identifier (LANGID) parameter and the sort sequence (SRTSEQ) parameter to access the unique-weight tables (*LANGIDUNQ) or the shared-weight tables (*LANGIDSHR).
- ▶ When using the relative sort sequence, the relative weights shown in these tables differ from the actual weights in the sort sequence table on the system. The relative weights shown in these tables are examples only.
- ▶ The relative unique weight of a character is shown by the order of the characters in the table. The relative unique weight is determined by assigning a weight of 1 to the first character in the table and incrementing by 1 for each of the following characters until the end of the table is reached.

Table 9-3 shows a subset of the table Common Sort Sequence for Latin-1. To see the complete table, go to:

<http://publib.boulder.ibm.com/cgi-bin/bookmgr/BOOKS/QB3AWC01/E.2.1>

Table 9-3 Latin-1 common sort sequence

GCGID	Character	Shared weight	Unique weight	GCGID	Character	Shared weight	Unique weight
SP100000	-	6	6	SM030000	&	47	48
SP080000	,	7	7	SM010000	#	48	49
SP140000	;	8	8	SM020000	%	49	50
SP130000	:	9	9	ND010000	1	64	69
SP150000	?	12	12	ND020000	2	65	71
SP060000	(27	27	ND050000	5	68	76
SP070000)	28	28	LA010000	a	73	81
SM060000	[29	29	LA020000	A	73	82
SM080000]	30	30	LA270000	ä	73	90
SM050000	@	37	37	LB010000	b	74	98
SC040000	¢	39	39	LB020000	B	74	99
SC030000	\$	40	40	LC010000	c	75	100
SM070000	\	46	47	LC020000	C	75	101

As Table 9-3 demonstrates, the shared weights of all forms of an alpha character are the same, for example, 'a' = 73, 'A' = 73, and 'ä' = 73. However, the unique weight is different, for example, 'a' = 81, 'A' = 82, and 'ä' = 90. With this in mind, review Table 9-4, which shows an example of a shared-weight sort sequence, a unique weight sort sequence, and the binary sort sequence for English code page 00037.

Table 9-4 Using the English language sort sequence table

Binary sort sequence	Shared-weight sort sequence using LANGID(ENU) and SRTSEQ(*LANGIDSHR)	Unique-weight sort sequence using LANGID(ENU) and SRTSEQ(*LANGIDUNQ)
Jones, Mary	JOHNSON, JOHN	JOHNSON, JOHN
JOHNSON, JOHN	JONES, MARTIN	Jones, Mary
JONES, MARTIN	Jones, Mary	JONES, MARTIN
Smith, Ron	SMITH, ROBERT	Smith, Ron
SMITH, ROBERT	Smith, Ron	SMITH, ROBERT

The iSeries ships with the QSRTSEQ defaulted to *HEX. The QNOTES user profile setting SRTSEQ defaults to *SYSVAL, but it is overridden by the LOCALE definition for *SRTSEQ, which is *HEX for all LOCALES used by Domino for iSeries. The display shown in Figure 9-1 demonstrates the user profile settings.

Change User Profile (CHGUSRPRF)		
Type choices, press Enter.		
Sort sequence	<u>*SYSVAL</u>	Name, *SAME, *SYSVAL, *HEX...
Library	_____	Name, *LIBL, *CURLIB
Language ID	<u>*SYSVAL</u>	*SAME, *SYSVAL...
Country ID	<u>*SYSVAL</u>	*SAME, *SYSVAL...
Coded character set ID	<u>*SYSVAL</u>	*SAME, *SYSVAL, *HEX...
Character identifier control	<u>*SYSVAL</u>	*SAME, *SYSVAL, *DEVD...
Locale job attributes	<u>*CCSID</u>	*SAME, *SYSVAL, *NONE...
	<u>*DATFMT</u>	
	<u>*DATSEP</u>	
	<u>*DECFMT</u>	
	<u>*SRTSEQ</u>	
	<u>*TIMSEP</u>	
Locale	<u>*SAME</u>	
User options	<u>*NONE</u>	*SAME, *NONE, *CLKWD...
+ for more values	_____	
		More...
F3=Exit	F4=Prompt	F5=Refresh
F24=More keys	F12=Cancel	F13=How to use this display

Figure 9-1 Work With System Status display

To force the QNOTES user profile to use the *LANGIDSHR sort sequence, enter the following command on any OS/400 command line:

```
CHGUSRPRF QNOTES SRTSEQ(*LANGIDSHR) SETJOBATR(*CCSID *DATFMT *DATSEP *DECFMT *TIMSEP)
```

This forces QNOTES to use the shared weight table that is specific to the language specified by the LANGID setting for the QNOTES user profile, and prevents it from using the *SRTSEQ setting specified by the LOCALE definition.

Note: The above setting has been tested and used when using Domino for iSeries and LEI for iSeries. It is possible to make the same changes to a user profile that is used to access DB2 UDB for iSeries data from a Windows NT-based server. However, this has not been tested.

It is possible to specify a specific table to use, and to even create a custom table. However, this is rare and should only be done by someone with considerable iSeries expertise.

In summary, by default, the iSeries is not configured to sort the data being accessed through the SQL Call Level Interface (CLI). This causes an order mismatch when retrieving data from DB2 UDB for iSeries tables via Lotus Domino, especially LEI and DECS.

By changing the QNOTES user profile, and configuring it to use a specific sort sequence, the issue can be addressed before it occurs. Because the data is presorted, there is no need to use an Order MetaConnector against a DB2 connection that references DB2, therefore, enhancing overall performance of LEI.

For more information, see the Lotus Technotes 184006 and 183813. You can find background information in:

- ▶ AS/400 National Language Support, SC41-5101
- ▶ AS/400 DB2 UDB for AS/400 SQL Programming Information, SC41-5611

- ▶ *AS/400 DB2 UDB for AS/400 SQL Reference Information*, SC41-5612
- ▶ *AS/400 DB2 UDB for AS/400 Call Level Interface (ODBC) Information*, SC41-5806

Other performance improvements for LEI

There are some things you can do to improve the performance of your LEI activities. The primary one is related to using the replication activity. If you are replicating larger quantities (more than 10,000 records) of DB2 and Domino records, we highly recommend that you use time stamp replication rather than primary key replication.

Let's look at a scenario to fully describe the performance situation of using primary key replication versus using time stamp replication. In the scenario, you are replicating 100,000 records in DB2 with 100,000 records in Domino. Only 18 of the records have changed in total.

- ▶ With primary key replication, all 100,000 records are examined in DB2, and all 100,000 records in Domino are examined. The 18 records that have changed are replicated to synchronize the databases.
- ▶ With time stamp replication, only the 18 records that have a time stamp greater than the time the last replication activity ran are examined. The 18 records that have changed are then replicated to synchronize the databases.

As you can see, there would be a dramatic performance difference between these two options. We have had customers that were performing primary key replication switch to using time stamp replication, reducing their replication times from 20+ hours to less than 45 minutes.

If used in the proper way, LEI is a great option for keeping DB2 and Domino databases in sync. However, if misused, LEI activities can be performance bottlenecks on your system.

@DB commands

@DB commands have been around for a long time in Domino and provide a very easy mechanism for accessing DB2 data from a Domino application. Because they are so easy to use, this programming technique often gets application developers into a lot of trouble. @DB functions are not very efficient. They are very CPU intensive, so when used inappropriately, they quickly create a performance problem.

@DB commands should be used sparingly in your application. If you have a Domino form that has multiple fields populated using @DB commands, you need to rethink your design. A form that contains multiple @DB commands to populate fields can take multiple seconds to load. This will cause unacceptable response times to end users. In addition, because these commands are CPU intensive, you will see spikes in the server CPU that may impact other work executing on the system.

We have seen applications in the past that used multiple @DB commands on a single form, causing the form to take over 17 seconds to load. By redesigning the form, the time to load the form was reduced to 0.5 seconds. This is a significant response time improvement. The changes involved in this performance improvement involved getting rid of unnecessary @DB commands by using cached data and moving some of the lookups behind buttons rather than having them evaluate in keyword fields and pop-ups. When formulas are placed behind buttons, they are not evaluated at document open time, therefore delaying the processing of these functions.

Note: When a Notes client opens a form with uses @DB commands to access DB2 data on iSeries, the PC needs to have an ODBC or DB2Connect definition to the iSeries server. If the forms are accessed by a Web browser or part of an agent running on the server, it uses Call Level Interface (CLI) to access the DB2 tables directly.

LotusScript Data Object (LS:DO)

LS:DO is a LotusScript Extension (LSX) that provides a set of classes for accessing backend databases and applications. This programming option is also very easy to use since only three different classes are involved in manipulating the data:

- ▶ ODBCConnection
- ▶ ODBCQuery
- ▶ ODBCResultSet

LS:DO connects Domino applications to data sources via ODBC technology. However, similar to the @DB functions, it uses Call Level Interface when accessed by a Web browser or part of a server-based agent.

Support for full read and write access to the external ODBC data source is provided. This option is quite popular because of its ease of use. However, beware of the performance issues you may run into.

LS:DO is not a great option when you:

- ▶ Need to access backend data frequently
- ▶ Retrieve large quantities of data

Lotus Connector LotusScript Extension (LC LSX)

The Lotus Connector LotusScript Extension classes are a set of LotusScript classes that allow you to write your own business logic to access backend data and applications. Using the LC LSX classes offers the ultimate in flexibility in designing an application. Not only is it supported on all Domino platforms, but the object model of LC LSX allows you to easily switch backend systems without having to change the application.

Coding your agent or script with LC LSX requires more skill than using the LS:DO set of classes, but the performance benefit is well worth it. The LC LSX classes allow you to perform very complex queries and return large quantities of data from DB2 to Domino while maintaining good performance.

Java Database Connectivity (JDBC)

Java is another programming language option when integrating Domino with DB2 UDB. Through the use of the Java Database Connectivity programming API, data can be retrieved from relational databases. Here you do not have the backend programming independence offered with the LC LSX classes, but you will find that using Java opens your applications to other programming options that out perform LotusScript programming options.

CORBA

The first programming option that Java opens our application up to is Common Object Request Broker Architecture (CORBA). CORBA is an industry standard specification that defines how distributed applications “talk” to each other. CORBA not only allows you to build robust, distributed applications for the Web, it also offloads processing from the server to the browser. Using the CORBA specification, remote Java programs can access Domino and DB2 server data as if it’s local. This means fewer trips to the server and less load on the server because some processing can be offloaded to the browser.

Servlets versus agents

The second options that opens when using the Java programming language is the use of servlets. Why is this so important? It has to do with scalability. The underlying Domino architecture creates a performance problem when we run agents. Agents are loaded from the Domino database when they are invoked. They perform the function they have been coded to handle, and then they are unloaded from memory. The very next time the agent is invoked, the same thing happens. It must be loaded into memory, execute, and be unloaded from memory. This is not very efficient!

Servlets on the other hand can be loaded when the HTTP task starts in the Domino server. Connections to backend systems can also be established at this point in time. As a result, when a servlet that has been preloaded into memory is invoked, it simply needs to execute. It does not need to be loaded and unloaded from memory. This is a *huge* performance benefit over using agents.

9.2.2 Connection pooling

Connection pooling is one of the items that is overlooked 98% of the time. It allows connections to backend resources to remain open and to be reused by subsequent processes that need to access the same resource. Without connection pooling enabled, when an agent or script code needs to perform a connection to a backend resource and retrieve results, the following steps occur:

1. A connection must be established to the backend system.
2. A full database open occurs against the backend database.
3. The query can be executed to extract the necessary results.
4. The backend database is closed.
5. The connection to the backend system is disconnected.
6. Results are returned to the application.

With connection pooling enabled, the following steps occur:

1. A connection request sees that an existing connection can be used from the connection pool.
2. The query can be executed to extract the necessary results because the database is already open.
3. The application disconnects and returns the connection to the pool of connections.
4. Results are returned to the application.

The main reason connection pooling is so often overlooked is that it is turned off by *default*. Unless you are using DECS or LEI, which use connection pooling by default, you need to explicitly turn connection pooling on when using one of the programming options discussed above.

Enabling connection pooling is very simple. It is a boolean value that is set to false (or turned off) by default. All you need to do is set it to true to enable it. For example, in the LC LSX set of classes, connection pooling is controlled by the ConnectionPooling property of the LCSession class. To turn connection pooling on for LC LSX, simply code:

```
Dim lcsession As LCSession
Set lcsession = New LCSession
lcsession.ConnectionPooling = True
```

It's that simple; just set the boolean value of ConnectionPooling property equal to *True*. The actual property that needs to be set varies with the programming language and option chosen. It is typically set as a property of the connection or session object.

By default, the maximum number of pooled connections for a data source is 20. You can override this default by adding the ConnectionPool variable to the server's notes.ini file:

```
ConnectionPool=oracle,10,db2,20,5,pssoft7,4
```

This example shows providing 10 pooled connections for Oracle, giving DB2 20 pooled connections with a maximum of five assigned to a specific data source, and PeopleSoft 7.0 gets four pooled connections.

Note: When using connection pooling, be aware that commits and rollbacks are not performed automatically when using commitment control because the file is never actually closed. The connection is simply returned to the pool of connections. You need to perform these functions explicitly in your code now.

9.3 Questions to consider when integrating Domino and DB2

With all of this information, let's look at some questions and answers that may be helpful in when designing your application. They give you a good idea of which of the integration options discussed will best meet your performance needs.

- ▶ Does the logic need to be inside the Domino application?
 - If not, consider using a DB2 function instead of a Domino application function. Many dynamic lookups are much better performers when executed by DB2; simply pass the query to DB2 for best performance.
- ▶ Does the information need to be stored in both environments, DB2 and Domino?
 - If not, DECS real-time access allows you to work with data stored in Domino document fields and at the same time display DB2 data for reading or editing. If many databases will be involved in this type of information exchange, look at writing your own business logic using LC LSX or Java.
 - If yes, LEI offers the best solution. This product has the built-in capability to synchronize disparate data sources. Keep in mind that LEI can cause performance bottlenecks when large quantities of records need to be replicated. Use time stamp replication when possible.
- ▶ How much data do you need to retrieve from DB2 and how often?
 - If you only need to retrieve a list of values, then @DB functions may be okay.
 - If you need to execute several statements with one SQL connection or work with large result sets, use:
 - LS:DO if the frequency of execution of the data retrieval is only every 2 to 5 minutes
 - LC LSX or Java using JDBC if the frequency of execution of data retrieval is 1 minutes or less
- ▶ Is your application Web-based?
 - Consider using Java servlets and CORBA. Servlets can be preloaded into memory for high performance. CORBA allows browser applications to interface with server-based data locally, without involving the server in each transaction. This not only cuts down on server CPU processing, but also improves network performance.



Clustering and partitioning

Clustering is designed primarily to provide high availability to the applications. In Domino clustering, the very fact that data is being physically duplicated in almost real time (cluster replication) to additional databases will impact overall performance. This performance impact will be greater if the whole clustering design is not carefully thought out beforehand and if it is not carefully monitored on an on-going basis once implemented. On the other hand, using the workload balancing feature to distribute the workload across multiple clustered servers may well improve overall performance significantly, by balancing all servers with an (more or less) equal workload.

Partitioning for Domino is the ability to run multiple Domino servers on a single iSeries server within an OS/400 subsystem. Domino partitions help to avoid contention within any one Domino server, particularly if there are too many users accessing that server at the same time. Because adding extra partitions (iSeries subsystems) significantly increases the demands on the server, you need to review resource allocation and possibly need additional hardware resource to maintain service levels and response times.

This chapter attempts to contrast the pros and cons from a performance point of view of partitioning and clustering.

10.1 Domino partitioning

Domino server partitioning is a feature that allows multiple Domino servers to run under a single instance of the OS/400 operating system, within a single physical machine or within one LPAR (see next paragraph). Each Domino partitioned server runs in its own OS/400 subsystem, and all Domino partitions share a single version of the Domino program code. You cannot have multiple releases or language versions of Domino running an single instance of OS/400.

Do not confuse the Domino partitioning concept with the iSeries logical partitioning (LPAR). The LPAR architecture allows you to run multiple operating systems (currently OS/400 or Linux) on the same physical iSeries hardware. Each LPAR can have a different version of the operating system or a different national language version and a different version of Domino with multiple Domino partitions. Each secondary LPAR can perform an IPL without interfering with any other LPAR.

The term “Domino partition” is also used generically to describe the Domino partitioned server across all Domino supported hardware platforms.

10.1.1 Why partition Domino on iSeries servers

The key goals for any Domino application running on a Domino server in an OS/400 subsystem is to improve both the reliability and the performance of the application. A Domino application running in one OS/400 subsystem will not be affected if an application in another OS/400 subsystem fails. This is actually a feature of the OS/400 job structure rather than the subsystem structure, which mainly groups the jobs into convenient manageable groups.

All OS/400 subsystems share and make the most efficient use of common resources such as the underlying disk subsystems, LAN and I/O cards, and CPU resources. By default, memory is shared between subsystems. But you can also assign a private memory pool to a single subsystem (not recommended) or use a shared memory pool by one or more subsystems (see 6.4, “Choosing which memory pool to use” on page 166).

The Domino servers do not share data between each other except by replication.

Before the advent of multiply partitioned servers, it was necessary to have a collection of small disparate Domino servers, each of which needed to have its own individual and separate software and hardware maintained. Today iSeries servers have been tested to run up to 32 partitions, with a theoretical limit of 99 partitions per OS/400. By also configuring multiple LPARs, the maximum number of defined Domino servers is limited only by the power of your iSeries hardware.

You can thus take full advantage of the iSeries inherent hardware reliability and highly available and scalable disk subsystems and combine all those small independent servers into one single physical box. This means there is only one physical machine and operating system to manage (within a single LPAR) and one version of the Domino software shared between all Domino servers.

There are also advantages to partitioning when the Domino servers are in the same domain. Even though there is some extra overhead in managing multiple replicas of Domino Directory address books (one per server) in this environment, there are distinct performance advantages to be gained from running several partitioned servers on one large single box, rather than on one single large partition. The deliberate distribution of the Domino server tasks, databases, and users across multiple Domino partitions on a single physical box provides the following key advantages:

- ▶ Better processing of multiple simultaneous requests for Domino services
- ▶ Maximized parallel task execution by the highly optimized iSeries operating system
- ▶ Smaller mail delivery lists by having multiple small Mail servers rather than one large one
- ▶ Faster mail.box access because less users share it and you have multiple mail.boxes
- ▶ A better distribution of the overhead of processes that must touch every database
- ▶ No (physical LAN) network to traverse for server-to-server traffic
- ▶ Instant reaction by Domino partitions and iSeries subsystems to initiate failover

There are limits on the number of users that can be handled by a single Domino server while maintaining good stability. In fact, it is *not* the number of users that actually matters. It is merely the workload those users are generating, how much resource it uses, and therefore, how much contention it causes. However, in an environment like a Mail server, where most users behave in a similar way, you can use the number of mail users to limit the workload per server. This cannot so easily be applied to any other Domino applications. This maximum number of users will vary from installation to installation. However, a good starting point or guideline would be to use the IBM Rochester's own mail server site guideline of 1500 to 2000 registered mail users.

If you have more mail users but they are light users, you may be able to increase the number of mail users supported on a single partition. Conversely if you have a lot of "power users", you may need to decrease the count on some servers, but we think that guideline is an excellent beginning reference point. See 7.10, "Number of users per Domino server" on page 201.

10.1.2 How Domino partitioning works

The Setup program automatically assigns a unique subsystem name (DOMINO01, etc.) to each Domino partitioned server. Or if you prefer, you can select your own name (up to 10 characters), which it must be unique on this OS/400.

When you configure the Domino server with Configure Domino Server (CFGDOMSVR) command, you can specify the desired name with parameter SBS or leave the default *GEN. This name is also used for some other OS/400 objects that are used for the server. These objects include a job description (*JOBQ), a job queue (*JOBQ), a class (*CLS), and possibly BRMS lists.

If a partitioned server encounters a fatal error, an automatic cleanup procedure allows the server to restart without interfering with any other subsystems or Domino partitions. This is accomplished by the QNNINSTS job, which permanently monitors all other jobs (that is Domino tasks) running in the same subsystem. If one of them fails, QNNINSTS gracefully brings down all other tasks and starts the entire server again.

Note: For automatic cleanup and restart to occur, the notes.ini file for each partitioned server must contain the KillProcess=1 setting.

The Domino program files are held in an OS/400 library (QNOTES) that is shared between all the partitioned servers, each running in its own subsystem and a data directory of its own. This is why all partitioned Domino servers run the same revision of the Domino code. This is slightly different to many iSeries applications where it is usually possible to run multiple language versions or releases under the same OS/400. Domino is different for various technical, internal, and optimization reasons.

Partitioned Domino servers can even share a single network adapter, either through port mapping or multiple IP addresses.

Port mapping allows the partitioned server to use the same IP address for communication, each server listening on a TCP port of its own. One of the servers has to be designated as the *port mapper*, who listens to the standard Domino port 1352 and reroutes requests to up to four other servers.

More flexible is the possibility to use *multiple IP addresses*. You can assign up to 2048 IP addresses to a single network adapter and up to 16384 for the entire iSeries server. Most customers, however, prefer that each Domino partitioned server has its own IP address.

Note: By default, a Domino server listens to all IP addresses on the system. If you plan to use partitioned servers, you must assign one (or more) IP addresses specifically to each server in the Configure Domino Server (CFGDOMSVR) command.

You can find full instructions for setting up both alternatives in *Lotus Domino for AS/400 R5: Implementation*, SG24-5592.

10.1.3 Planning for Domino partitioning

When you begin the planning phase in regard to partitioning, consider the following questions:

- ▶ How do partitioned servers fit into your current Domino environment?
- ▶ How many partitioned servers should you install?
- ▶ What are the unique IP address considerations for each server?
- ▶ Rules for naming each Domino partitioned server?
- ▶ Rules for naming the data directory of each Domino partitioned server?

Before starting to create your partitioned Domino servers (which also automatically creates the OS/400 subsystems at the same time) on your iSeries server, be sure you perform the following actions:

1. Select option **7** (Advanced Services) when you run the Load and Run (LODRUN) command.
2. When you use the Configure Domino Server (CFGDOMSVR) command to set up a Domino server, specify either **ALL* or **PARTITION* for the Advanced parameter. In fact, you should set this even for the first server. Admittedly at that point, it cannot be called “partitioned”, because there is only the one. But, if you decide at a later point to add Domino partitioned servers, you have to change the first one with the Change Domino Server command, if it was not set up partitioned in the first place.

When you run the CFGDOMSVR command, if it fails with the message LNT0109 (Maximum number of servers exceeded), this usually means you did not configure partitioning support for one of the servers (which may be the existing server).

To tell if an existing server was configured as partitioned, check the server notes.ini file for the variable NPN=1. This is the first partition.

Note: In Domino R4, you could check for the existence of partitions by checking the existence of file /QIBM/UserData/Lotus/Notes/lotus_servers/.sgf.notespartition. After Domino R5.02, this file was no longer used and now there is a file called /QIBM/UserData/Lotus/LOTUS_SERVERS.

A new function was added to the QNNINAPI service program (Get_Partition_Number API), which is what the installer now uses to check Domino partitions.

Once up and running, it is important to remember to look at system resources along with Domino statistics. You should regularly determine which Domino partitioned server is using the most and the least system resources. That is to say all of the partitioned servers should be monitored as a group to ensure that the workload is reasonably evenly balanced between the servers. If one server is extremely over loaded, this will affect other servers that rely on that server for part of their own performance.

For example, the user of a mail server will perceive poor mail delivery performance if their outbound mail is queueing waiting on an overloaded SMTP server in another subsystem. The affect of one server on the performance of another server has to be taken into consideration and regularly monitored or at least checked!

Monitoring a partitioned server is no different from monitoring a single Domino server. Partitioned servers include the same set of tools as a single Domino server. What is important to remember with partitioned Domino servers is that a particular Domino server may take advantage of any shared system resources or pools and could deny that resources to another partitioned Domino server that shares the same pools and resources.

10.1.4 Partitioning tips

When running Internet tasks, such as HTTP, POP3, IMAP, and SMTP, on Domino partitioned servers, you need to perform additional configuration steps if you intend to run these services on more than one Domino partitioned server, or you are also running iSeries native versions of HTTP, SMTP, and so on. For example, if you are using a unique IP address for each server, you need to bind that hostname with the HTTP server. If you don't do this, any other HTTP, SMTP, and similar "service providers" that start up in the future or that are already running on the same iSeries server might try to use the same host name, and a conflict would occur. The binding is done by editing the server document and setting the host name value in the HTTP server section of the server document.

Managing partitioned servers involves the same concepts as managing any Domino server or any iSeries subsystem running any production application. Each Domino partitioned server has its own private administration interface through the Work with Domino Console (WRKDOMCSL) command or option 8 of the Work with Domino Servers (WRKDOMSVR) panel. The console can also be accessed from a *Domino administrative client* on a Windows workstation, which in turn can be activated from the Domino functions in Operations Navigator.

You use the same set of tools to analyze Domino partitioned servers that you use to analyze the activities of a single Domino server. The main difference is how the servers are monitored. You have to remember that the partitioned servers share the resources of the same physical machine. Therefore, bad server performance on one partitioned server could be the result of a task on another partitioned server. When analyzing server performance, you have to look past individual server statistics. Look at system resources and compare the statistics for each individual server

Before you start any of the partitioned servers for the first time, the notes.ini file needs to be modified to define which IP address and port each server is listening to and if a port mapper should be used. Starting with R5, this can be done during the configuration with:

```
CFGDOMSVR .... TCPOPT(*NOENCRYPT '10.1.2.3')
```

From the Notes client workstation point of view, two partitioned servers look like two different systems, that is, two different IP hosts. From the iSeries server TCP/IP configuration point of view, there are two different ways to implement partitioned servers:

- **Unique IP addresses**

If the servers use the same iSeries communication adapter, you need to assign more than one IP interface to the same line description. Each server must appear with its correct IP address on a name server (DNS) or the client host table.

- **Share IP addresses using a Port Mapper**

All servers running on the iSeries server use the same IP address. Unique port numbers are used to distinguish between each partitioned server.

Once you configure multiple Domino partitioned servers on the same iSeries server, you may need to replicate your databases or allow mail routing between those servers. Since these servers reside on the same iSeries server, it is desirable to exchange information between them without the need for an external communication network. Domino for iSeries servers use TCP/IP for replication and mail routing.

10.1.5 Communication between Domino servers on the same iSeries server

In the past, we recommended that you define a *loopback interface* for the communication between servers on the same AS/400 hardware. This is a special virtual TCP/IP interface that allows socket connections without the use of a real hardware interface. The TCP/IP address of the loopback interface is 127.0.0.1. It is defined on every system, using TCP/IP as a networking protocol. If you set up two or more partitioned servers either with unique IP addresses or using a port mapper, we recommended in the past that you perform additional configuration to allow those servers to connect using the loopback interface.

OS/400 V4R4 and later realizes when two IP ports are on the same system and then allows them to communicate without even involving any communication interface. Therefore, it is no longer necessary to use the loopback interface. In V4R4, TCPIP was enhanced so that local IP addresses are handled internally by the TCPIP stack. As a result, it is better to not configure one manually (for example, `PORTS=TCPIP,LOOPBACK` in `notes.ini`). If you take it out for OS/400 V4R4 and later, you simplify administration and eliminate some Domino overhead.

The difference from V4R3 and V4R4 concerns sending from one local IP address to a different local IP address. With the latest cum tapes applied, the current implementation is:

- **V4R2 and earlier:** Packets sent to any non-`*LOOPBACK` local IP address are sent out on the wire.
- **V4R3:** Packets sent to and from the same local IP addresses are looped internally, which is similar to `*LOOPBACK` handling.
- **V4R4:** Packets sent to and from any local IP addresses are looped internally, which is similar to `*LOOPBACK` handling.

With no PTFs applied, the base code of V4R3 code worked like V4R2 and the base code of V4R4 worked like V4R3. But the PTFs to enhance local packet handling have been on V4R3 and V4R4 cum tapes for some time.

10.2 Clustering

The iSeries itself has a number of clustering architectures, which we highlight here to avoid confusion and show the various options. There has been Domino clustering (since Domino 4.x), and now we have also iSeries and OS/400 based cluster functions. All can be used independently or in combination.

10.2.1 iSeries clustering

Clustering is employed by the iSeries to provide availability protection from both planned and unplanned outages. IBM has spent a significant amount of design and development resource to enable high levels of availability in a single system environment. It is becoming increasingly clear that there is no time for either planned or unplanned outages. Unlike many platforms, the iSeries uses its single system high availability, high reliability design for protection from unplanned outages. Clusters are deployed to offer protection from planned outages and site disasters as well as other unplanned outages. In the iSeries world, the lion's share of outage time comes from planned outages. By solving the planned outage problem, the unplanned outage coverage is a side benefit.

iSeries continuous availability

To further address the needs of the continuous availability market, iSeries has invested in cluster technology. Significant advanced features and functions were introduced in V4R4 that put iSeries in a leadership position. Cluster technology has been around for many years, but only recently have common terms and concepts begun to emerge. In the case of iSeries, it has adopted these concepts and integrated the infrastructure needed to achieve a level of standardization, an advancement of cluster technology, and the involvement of the Solution Developer (SD) and the cluster middleware business partner in the total solution. While the iSeries provides the basic cluster infrastructure, the cluster middleware business partner provides data resiliency and the ISV provides application resiliency. Together, these three parts provide the whole solution to continuous availability.

iSeries solutions for the continuous availability market

Beginning with OS/400 V4R4, cluster technologies provide our cluster middleware business partners, solution developers, and customers a set of key integrated functions for building solutions for the continuous availability market. Already independently rated as the most highly available single-server platform on the market today by a Gartner Group study, the iSeries cluster architecture delivered beginning with V4R4 allows application developers to build continuous availability solutions that handle both planned and unplanned outages.

Synchronizing and managing multiple iSeries servers

iSeries integrated cluster technologies provide the key functions and interfaces necessary to synchronize and manage multiple iSeries servers ... all the while monitoring for system failures and providing key data and application switchover administration. In addition, new IP address takeover functions further simplify this support, delivering high availability Internet serving capabilities. With this technology, and in cooperation with cluster middleware business partners, the iSeries server is further enhanced with easy-to-use cluster management and robust data resiliency required in today's 24x365 e-business world.

Integration of IBM Business Partner cluster management software

Cluster Resource Services, included in OS/400, enable IBM High Availability Business Partners' cluster management software to easily create, configure, and administer iSeries clusters. With an architecture that supports up to 128 servers and a choice of OptiConnect, ATM and LAN connections in any combination, customers can tailor iSeries clusters to meet the availability needs of their business.

Highly available applications using a common interface architecture

Solution Developers can build enhanced high availability applications using a common interface architecture to cluster management and data resilience products. Properly designed, these applications can be configured and activated from cluster management utilities and made resilient to planned and unplanned outages. Applications meeting the criteria may receive the right to be recognized as ClusterProven™ for iSeries. The Cluster Middleware solutions from IBM Business Partners support ClusterProven applications.

For more details, see the high availability Web page at:

<http://www-1.ibm.com/servers/eserver/iseries/ha/haclusterwebpage.htm>

OptiConnect for OS/400

OptiConnect for OS/400 provides solutions to many high availability and capacity problems by enabling capacity growth through shared database clustering. The OptiConnect cluster not only achieves horizontal growth and high availability but also aids in data warehousing and database parallelism architectures.

The OptiConnect cluster consists of a collection of systems, each of which dedicates a system bus to connect to a common or shared bus. The system that provides this shared bus is referred to as the *OptiConnect hub system*. The systems that attach to this shared bus are referred to as *OptiConnect satellite systems*.

The OptiConnect system area network connects multiple cluster nodes, either unique systems or partitions, using one of the three high-speed technologies:

- ▶ **HSL OptiConnect:** A system-to-system connection that uses the High Speed Link (HSL) technology.
- ▶ **Virtual OptiConnect:** A high-speed communication connection between partitions. In addition, a partition can participate in an HSL or SPD OptiConnect connection.
- ▶ **SPD OptiConnect:** Provides high-speed connections between multiple systems using SPD OptiConnect hardware.

For more details, see the OptiConnect Web page at:

<http://www-1.ibm.com/servers/eserver/iseries/opticonnect/>

10.2.2 ClusterProven Domino for iSeries

ClusterProven Domino for iSeries is an iSeries-specific enhancement for Lotus Domino that allows a V5R1 OS/400 cluster management (CM) utility to manage iSeries-based Domino servers. In the case of ClusterProven Domino, the “application” that is managed through OS/400 cluster management functions is an instance of a Domino server defined to an iSeries server that is also configured as a cluster node. Any number of Domino servers can be configured on a cluster node and each Domino server can further be individually configured and defined to the cluster as a part of an application *Cluster Resource Group (CRG)*. A list of cluster nodes is configured as part of the CRG. The Domino server can be started on any of these nodes and is automatically created on all these nodes when the application CRG is created and the Domino server is first started on each node.

Automatic failover

Using cluster management to support Domino provides automatic failover to backup iSeries servers or logical partitions (LPARs) in the case of a system failure. Perhaps even more important than the built-in support for failover provided by OS/400 cluster management is the support for a function known as “switchover”. Switchover entails essentially the same functions that allow a ClusterProven application to be restarted in the case of a failover but enables operational control so that the process of ending the application on one cluster node and restarting it on another can be done “on demand”.

In the case of a Domino server configured as a cluster resource, switchover can be used to effectively switch the same Domino server so that it can be ended on one system and restarted another. This would be an ideal means to provide continuous support for the server's applications and users on a backup system in the case of system downtime for scheduled maintenance or a system IPL.

Defining and starting Domino on multiple nodes in a cluster

The means of allowing the same Domino server to be defined and started on multiple nodes in a cluster without maintaining multiple copies of its data on these nodes is provided via an independent auxiliary storage pool (IASP). In OS/400 V5R1, these IASPs can be defined so that IFS file object access can be switched between two iSeries servers.

ClusterProven Domino represents the first practical example of how an application CRG can provide “application resiliency” using OS/400 cluster services and have that application's “data resiliency” provided using a separate device CRG to manage the access to the application's data. In the case of a Domino server, the application data is essentially the server's entire data directory.

A device CRG allows the IASP to be managed as a cluster resource, most importantly it allows access to the same IFS file objects that are part of a Domino server's data directory to be switched between nodes in the cluster. In this case, the device CRG allows a single copy of a Domino server's data directory to be created within a user defined file system (UDFS) defined as part of the IASP device. In this way, access to a single copy of a Domino server's IFS file objects can be switched from one node in the cluster to another.

Switching and restarting Domino on different cluster nodes

The same “switchover” concept that allows the Domino server to be “switched” to be restarted on different cluster nodes is also used as the means to manage access to the server's data directory. In this case, the device CRG is what is switched between the cluster nodes. When the device CRG is switched to a new cluster node, the device associated with the device CRG is varied on and the UDFS associated with that device is automatically mounted within the IFS file system on that cluster node.

To a Domino server that has its data directory contained within that UDFS, the affect is that its data directory simply “appears” on that node and the Domino server can then be started on that node.

Automatic IP address takeover to a new cluster

Lotus Notes clients access their Domino server through a configured IP address. When a Domino server is configured with an application CRG, the server's IP address is used for the configured IP address of the server's application CRG as well. When an application CRG fails over or is switched over to a different cluster node, OS/400 cluster management automatically does IP address takeover to activate the configured application CRG's IP address on the new cluster node. This is ideal for Notes client because they automatically reconnect to the Domino server through the server's configured IP address when the server is restarted on the

new cluster node. There is no special configuration required on the Notes client. Except for the period of time that a Domino server is ended and restarted on a different cluster node, the user of a Notes client is not aware that a Domino server was failed or switched over and restarted on a different cluster node at all.

Managing the Domino server like any other cluster resource

When configured as high availability servers with an application CRG, a Domino server can be managed like any other cluster resource. As an application CRG, it can be started, stopped, or switched over between cluster nodes. This can be done with Operations Navigator Management Central, with OS/400 cluster management APIs and CL commands, or with high availability business partner applications that support the management of iSeries clusters and their application CRGs. The device CRG that controls the access to the Domino server's data directory can also be managed through these same interfaces.

ClusterProven Domino

ClusterProven Domino requires OS/400 V5R1 and Domino QMR 5.0.7 to be installed on every cluster node if that node will support Domino servers configured as part of an application CRG.

Some restrictions apply

Some restrictions apply in regard to the functions that a Domino server configured with an application CRG can support. The OS/400 system directory integration and using the OS/400's mail server framework to provide e-mail support should not be enabled for a Domino server that can be started on multiple cluster nodes. Other precautions have to be taken to support Domino server-based applications such as those that directly access iSeries DB2 databases or call iSeries program objects for example. Domino server-based applications that directly access system specific objects as part of their solution might not work in an environment where the same Domino server that supports the applications could be started on different iSeries servers within a cluster.

Domino clustering versus ClusterProven Domino for iSeries

Domino itself provides a Domino database clustering capability that is still fully functional in environments where Domino servers are configured as part of an application CRG. In the case of Domino database clustering, Domino databases are configured so that if a given Domino server fails, a backup server that effectively maintains a copy of that database will automatically be used to provide continued access to that database. This is specific to each database and requires a copy of the database to be stored and kept current on every Domino server that would provide support for the Domino database if the database's server was not available.

In the case of providing continuous availability through ClusterProven Domino for iSeries, instead of the a Domino database being replicated to multiple servers as a means to provide continuous access, the entire server is itself duplicated to multiple iSeries cluster nodes, each having the ability to start that same Domino server. Each of these cluster nodes can potentially access any Domino database that is defined within that server's data directory.

Effectively the databases are not replicated, but rather the server definition itself is copied to the multiple cluster nodes. Each node has the ability to have access to the same server's entire data directory through an IASP switched to that cluster node. That cluster node can start the same Domino server and access the same, single copy of its Domino databases maintained on an IASP that can be switched from node to node within the cluster.

10.2.3 Domino clustering

Domino clustering refers to a set of Domino servers that make up a loosely-coupled set of systems and application instances. To put it simply, Domino maintains replicas of either all or typically only selected databases on other Domino servers on a very frequent or even on demand basis.

Should any Domino server fail, the cluster management services (Domino Cluster Manager and Cluster Admin) redirects all subsequent client traffic to an alternative cluster replica, pretty much transparently to the user and with minimal data loss due to the crash. Another benefit offered by the Domino cluster management services is (optionally) load balancing the members of the Domino cluster to evenly spread the workload between servers

Domino cluster benefits summary

The advantages of Domino clusters can be summarized as follows:

- ▶ Provide load balancing and failover capabilities through event-driven replication.
- ▶ Provide uninterrupted access to Notes networked information resources, including Notes messaging, databases, and other service components.
- ▶ Provide higher availability and scalability, and enable more efficient use of resources than is possible from a single application instance system.
- ▶ Interconnects between cluster members within the same Notes domain across any network connection of LAN, ATM, or a high-speed WAN connection.
- ▶ On iSeries, allows multiple iSeries Domino partitions to be clustered within the same iSeries server for maximum availability, flexibility and load balancing.

The iSeries fully supports Domino clustering within the same iSeries server between multiple iSeries Domino partitions. Being “platform agnostic”, Domino servers can be easily configured to cluster with other iSeries servers, as well as with all the non-iSeries Domino server platforms. Because of the inherent reliability of the iSeries hardware, clustering within the one iSeries server is a key delimitator. It provides the failover support expected to protect key sites against the failure of tasks within Domino subsystems.

10.2.4 Why cluster Domino servers

This section explains the importance of clustering Domino servers.

High availability of critical applications

Critical server tasks such as mail, replication, and passthru tasks particularly benefit from operating in a clustered environment. For applications that require 7-by-24 availability, Domino clustering can help by enabling multiple replicas of these applications to be installed on a Domino cluster using anywhere from two to six Domino servers. If a server goes down, the user is redirected to another available Domino server that houses the application requested by the user. Therefore, users accessing these applications can achieve high availability. Note these benefits:

- ▶ With failover, users can continue accessing their critical data even after a server becomes unavailable. When a server goes down, Domino clustering and the cluster-aware Notes client help redirect user requests to the best available data source in the cluster. This provides failover protection for business-critical databases and servers including pass-through servers to other servers in a cluster format, and Domino Web users such as iNotes Web Access, iNotes for Outlook, and QuickPlace.
- ▶ The workload balancing feature lets you distribute user workloads across multiple clustered servers. You can create replicas of heavily used databases on other cluster

members, which will ensure that heavily used servers can pass requests to other cluster servers and that work is evenly distributed across the cluster. Through the use of Domino load balancing, users are dynamically redirected to the lesser loaded server to allow for more efficient resource utilization in a cluster. Workload balancing uses the same techniques as failover, but with the server availability threshold parameter. If the server is available and it has been determined that the server is too busy to take on any more user or server sessions, the administrator can redirect users to a more available (less heavily loaded) server. Workload balancing appears as virtually transparent to the user. If they are switched to a different server, the user will just see an additional icon on the desktop or a stacked icon for the database on the second server. They will not receive an error message or other indication that workload balancing has occurred.

Scalability

As the number of users, transactions, and applications increase, you can add servers to your cluster. By adding replicas of critical databases throughout your cluster, you can distribute the workload to optimize system performance. This ensures efficient resource utilization and maintains a high degree of scalability to provide flexibility for future growth.

Flexibility

There is no requirement that each clustered Domino server run on the same operating system or hardware platform. This advantage of great flexibility allows Domino clustering to assist in server consolidation and software migration. On iSeries, Domino clustering is frequently used to the same iSeries server because of its inherent reliability to achieve load balancing and maximum availability.

Disaster preparedness

Domino clustering can be seen as an enhanced backup feature when developing disaster preparedness plans. By using Domino clustering locally or across a WAN to a Domino server in another location, the clustering of mission-critical information provides a replica of the databases you want backed up to a safe backup or "off-site" location in almost real time.

10.2.5 How a Domino cluster works

In the event of failover or load balancing, the users of the failing server will "failover". That is they will be transferred to other servers in the cluster. The users return to their home server the next time they start the client, provided the home server is available. Notes clients are re-directed by the Cluster Manager, and Web clients (browsers) are directed and re-directed by the Domino Internet Cluster Manager (ICM).

Event-driven cluster replication

Domino clustering provides event-driven cluster replication among servers in the cluster to keep tight synchronization of data across the cluster. Within a cluster, every database change is replicated online. Every time a document or any of its properties is changed, that change is replicated to all of the database replicas within the cluster. The Domino Directory (formerly known as the Name and Address Book or NAB) is where clusters are defined and servers added or delete from that cluster. When you add a server to a cluster, the Administration Process (ADMINP) adds the cluster name to the server document.

Notes client cluster components

The Notes client is cluster-aware and maintains the cluster cache (an image of where all the current cluster components are) locally. The cluster cache provides a list of servers in the cluster. When the client tries to open a database on a Domino server that is unavailable, the client uses the cluster cache to find the next available server in the cluster. There are two Notes client cluster components:

- ▶ **Cluster-aware Notes client:** When the Notes client tries to open a database on a Domino server that is unavailable, the cluster cache (CLUSTER.NCF file in the data directory) on the client provides a list of other servers in the cluster. The Notes client accesses the cluster manager on another server in the cluster, which redirects the client to the best available Domino server that holds the content (replica) requested by the Notes client.
- ▶ **Cluster unaware browser client:** When a standard desktop Web browser tries to access a URL (that is really a Web enabled Notes database on a Web-enabled Domino server), the TCP/IP request is re-directed to the best available Domino (Web) server that holds the content (replica) desired by the browser. If that Domino server then fails, on the next HTTP request (refresh, next page, next selection, etc.), the ICM will catch the request, and re-direct it to an alternative Domino (Web) server that holds another (replica) of the database requested by the browser.
- ▶ **Notes API:** Notes APIs exist to allow Notes applications to be written that take full advantage of Domino failover and load balancing. The Domino server, Notes client, and Web browser work together to provide all the clustering features, so no setup at all is required on the client or in the browser, making your clustering rollout extremely easy.

Cluster Administration Process (CLADMIN)

This server task is responsible for the correct startup of all cluster components. On clustered servers, the process runs automatically at server startup and whenever the cluster membership changes. CLADMIN is a Domino add-in task. It is responsible for the correct operation of all cluster components, which include the configuration (adding or deleting) and launching of other cluster components.

Cluster Manager

Cluster Manager runs on each server in a cluster and tracks the state of all members in a cluster. The Cluster Manager is a ClusterTask thread in the Domino server and is created at server cluster startup or when Domino detects a change in NAMES.NSF. Information about the servers in the cluster is stored in the cluster cache on each server in the cluster.

Cluster Manager tracks all members in a cluster by:

- ▶ Determining which servers belong to the cluster and monitoring the Directory for changes
- ▶ Polling the other servers in a cluster and exchanging periodic messages (called *probes*) and keeping a list of which servers in the cluster are currently available
- ▶ Advising other Cluster Managers of changes in cluster server status
- ▶ Redirecting database requests, based on known cluster server status
- ▶ Balancing server workloads in the cluster, based on known cluster server status
- ▶ Logging failover and workload balancing events in the server log file

Cluster Replicator (CLREPL)

The Cluster Replicator is a Domino add-in task. CLREPL uses the cluster database directory to determine databases changes. It is responsible for:

- ▶ Tightly synchronizing data among databases and their replicas in a cluster
- ▶ Providing continuous event-driven replication among the cluster members
- ▶ Pooling changes within the replicator to provide improved performance

Cluster statistics cache

The cluster statistics are stored in the cluster statistics cache on each Domino server in the cluster. There are two kinds of cluster statistics:

- ▶ **Server.Cluster:** These statistics are related to clustered server activities such as failover, workload balancing events, the server state within the cluster, and information on a specific cluster configuration such as the server names within the cluster.
- ▶ **Replica.Cluster:** These statistics are related to cluster replication events, such as the number of documents updated, number of replica retry events, and the number of bytes received during cluster replications.

Cluster Database Directory (CLBDIR.NSF)

The Cluster Database Directory is updated by the Cluster Database Directory Manager and is kept in synchronization by the Cluster Replicator. The CLBDIR.NSF replica ID number is stored in the server document of each server in the cluster. The database document contains information such as the database name, server, path name, replica ID, and other replication and access attributes. It is common to all of the clustered servers for the use of determining failover paths and access control. It provides other cluster tasks and components with the data that they need to perform their functions. And, it manages databases with cluster-specific attributes such as databases marked out of service or pending delete.

Domino servers fail over clients by redirecting database requests to other servers in the cluster. This redirection is managed by a Cluster Manager.

Through the use of Domino workload balancing, users are dynamically redirected to the lesser loaded server. Workload balancing uses the same techniques as failover. However, the administrator sets the level when the server is considered to be too busy, and new client requests are “failed over” to another server that is not as busy as the original server. Workload balancing depends on the server state. The server state (busy or available) is determined by the server availability threshold value. When the server availability index drops below the server availability threshold, the server becomes busy. If a client tries to open a session with a request that triggers failover, they are redirected to an available clustered server. The workload on the server is expressed as the server availability index, which is a value between 0 and 100. The value 100 indicates a lightly loaded server (fast response times), and the value 0 is a heavily loaded server (slow response times).

Despite the fact that the server availability index is a number between 0 and 100, it is not a percentage. The server availability index is closely related to a common performance metric called the *expansion factor*. The expansion factor is simply the ratio of the response time for a function under the current load to the response time for this same function in an optimum (light load) condition. To compute the server availability index, the Domino server computes the expansion factor for a representative set of Notes RPC transactions over a recent time interval (roughly the previous minute). The server availability index is then set to 100 minus this expansion factor. Remember that the server availability index only considers the response time as measured at the server, which is typically only a small portion of the overall response time as seen by clients. In particular, the network time between the client and server often accounts for a significant portion of client response time.

10.2.6 ICM and the Domino HTTP Clustering components and technology

Domino R5 supports failover and workload balancing of HTTP and HTTPS client access to the Domino HTTP servers running in a Domino cluster. This capability is provided by a Domino application called the Internet Cluster Manager (ICM).

Running as a Domino server task, the ICM serves as a liaison between the HTTP clients and the HTTP servers of a Domino cluster. The HTTP clients direct requests for a database to the ICM. The ICM maintains availability information of the Domino servers in the Domino cluster and maintains information about the distribution of databases on the servers. The ICM determines the best server to receive a particular client request and directs the request to that server.

In summary, ICM and Domino HTTP Clustering offer these functions:

- ▶ Monitors backend Domino servers and HTTP task for availability
- ▶ Disallows any new connections to servers out of service
- ▶ Fails over new connections to the best available server
- ▶ Gives you the ability to set priorities (availability thresholds) for your Domino servers
- ▶ Provides content routing for your clients
- ▶ Failover and load balance over Web sites that contain different content

ICM corporate and enterprise use of Domino HTTP Clustering

Domino HTTP Clustering provides failover and load balancing of browser client access to Domino applications. An organization that wants to provide high availability to its customers that use browsers to access Domino Web applications would enable HTTP clustering to provide failover support for their Domino servers. An organization that wants to provide scalability for its Domino Web applications would enable HTTP clustering to achieve load balancing of browser clients that access their Domino Web applications. Domino HTTP clustering finds the best available Domino server and redirects the browser client to that server.

ICM characteristics of Domino HTTP Clustering

The characteristics of Domino HTTP Clustering are end-to-end security and authentication. Secure Sockets Layer (SSL) encryption and decryption are preserved. If you choose to require name and password and SSL support on your servers, the ICM preserves those capabilities. Domino HTTP Clustering also provides simple but strong affinity of the client with the server. The ICM does not require any additional hardware but can be installed on your Domino server.

ICM platform support for Domino HTTP Clustering

The ICM runs on all the operating systems and hardware platforms supported by the Domino server:

- ▶ Windows NT
- ▶ Sun Solaris
- ▶ HP-UX
- ▶ Linux
- ▶ iSeries
- ▶ pSeries
- ▶ zSeries
- ▶ xSeries

Domino HTTP Clustering is available for any of these platforms, and the servers in the cluster may be any combination of these platforms.

ICM network protocols supported by Domino HTTP Clustering

Domino HTTP Clustering supports client connections via TCP/IP only. Any supported Notes protocol can be used between the ICM and the servers in a cluster.

ICM conditions triggering failover and load balancing to occur

The ICM conditions that trigger failover and load balancing are:

- ▶ The Domino server is unreachable due to hardware, software, or network failures.
- ▶ The Domino server is restricted.
- ▶ The Domino server is busy.
- ▶ The Domino server reaches the maximum number of user sessions allowed as specified by Server_MaxUsers setting.
- ▶ The Domino database (replica) is unavailable because it is either marked out of service or marked pending delete.

ICM workload balancing

Workload balancing is enabled on the Domino server by using the setting Server_Availability_Threshold in the notes.ini file. If the server availability index falls below the specified threshold, then the server is busy. The ICM dynamically load balances the workload to a better available server.

ICM server availability index and maximum scalability

The server availability index is a value between 0 and 100 that indicates the workload on the server. It is calculated based on the response time of a representative set of server operations that takes into account both Notes clients and Web client access on the Domino server. The article “Workload Balancing with Domino Clusters” on the Iris Today Web site explores these questions in detail. You can find the Iris Today Web site at:

<http://www.notes.net/today.nsf>

ICM and user identification plus authorization

The security model for the ICM is to support the current Domino security model used for Web access. The HTTP requests are sent with no user ID or password to the ICM. The ICM then redirects requests at the protocol level and sends the redirection response code and target server information directly back to the HTTP client. The HTTP client then issues a request to the specified target Domino server whereupon the local Domino security policy takes effect, which may result in authentication dialogs between the HTTP client and the target Domino server. If the user successfully authenticates, the user ID and password are supplied in the HTTP headers for all subsequent requests to that Domino server. In this manner, the ICM is not involved in the user identification/authentication dialog.

ICM and SSL

The ICM uses SSL for transport level security. The ICM is configurable to require SSL sessions to be used. The ICM can use the same certificates that the browser and Domino HTTP server use.

ICM unauthorized access

Install an Internet firewall when it is necessary to protect the ICM from unauthorized access.

ICM and bookmarks with Domino HTTP Clustering

If a browser creates a bookmark, it will contain the URL of the backend Domino server that contains the page that is being viewed. The bookmark will not point to the ICM, and therefore, it will not support failover at this time.

ICM URL processing

The ICM accepts and processes all URLs currently supported by the Domino HTTP server, including Domino URLs to:

- ▶ Open servers, databases, and views
- ▶ Open forms, navigators, and agents
- ▶ Open, edit, and delete documents
- ▶ Open documents by name from a view
- ▶ Open image files, attachments, and OLE objects
- ▶ Create search queries

ICM non-targeted requests

In the current version of the ICM, when an HTTP client makes a request to the ICM that is not targeted at a Domino database, the ICM simply selects the most available server in the cluster that is running a Domino HTTP server and redirects the HTTP client to that server. Files not stored in a Domino database should be replicated across all servers in the cluster.

ICM and Domino

Changes were made in the Domino HTML generation to direct HTTP clients to the ICM for databases within a cluster if that cluster is configured with ICM support. The ICM knows the location of databases because the ICM has access to the CLDBDIR.NSF, which houses information about all the databases in the cluster, about the servers on which they reside, and database availability indicators.

ICM Domino HTTP Clustering performance

Domino HTTP Clustering provides for failover and workload balancing of HTTP and HTTPS client access to the Domino HTTP servers running in a Domino cluster. Users that access Domino servers through the ICM can experience some small increase in response time on the first access to a Domino application. This increase in response time is caused by the extra network communication needed for the redirection. Once redirection has occurred, the user communicates directly with the target server, so response times will be identical to a configuration without the ICM. Furthermore, because the ICM distributes the total client load across the servers in the cluster, overall client response times should improve.

Performance testing revealed a performance advantage to having the ICM in the cluster, in that a redirection to the HTTP task on its server will have slightly better performance. This is because redirection to the same machine can reuse certain low level network connections and therefore be more efficient. The disadvantage to this approach is that the server will have the additional overhead associated with executing the ICM tasks.

10.2.7 Planning for Domino clustering

When planning for Domino Clustering, examine the following list to ensure that all of the components and the effects of the clustering process are given some consideration:

- ▶ The advantages that you want to gain from Domino clustering
- ▶ How the organization is distributed geographically
- ▶ Networking considerations
- ▶ Number and distribution of database users
- ▶ Expected volume of new data, records added, updated or deleted per day, week or year
- ▶ The overhead of the cluster replication on each server
- ▶ Memory and CPU cycles for the cluster tasks
- ▶ Sensitivity of data
- ▶ The server roles
- ▶ Whether to use a dedicated server

- ▶ The number of servers to put in a cluster
- ▶ Plan the databases to be clustered
- ▶ The databases that need high availability
- ▶ Number and type of databases
- ▶ Size of the databases
- ▶ Load balancing
- ▶ Mail database integration
- ▶ Estimated growth

The cluster management tasks themselves have little impact on the performance of the individual servers, except for a minimal amount of server-to-server communication that is used to determine availability. The overhead of clustering on server CPU and resource workloads comes from the work and I/O required to actually replicate all the cluster member databases and data changes as they occur. Every server holding a replica of a database to be kept synchronized through Domino clustering creates a little more LAN traffic and small increases in the CPU, memory, and storage requirements. The cluster members should have enough spare capacity to reasonably service the users of any failed server. In the case where two existing mail servers are put into a new cluster of two, and all the databases are replicated onto both server, consider the CPU and resource utilization of two servers when suddenly added together and placed on just one of them. If these servers are primarily mail servers, then, when they are clustered, all the users' mail databases will be replicated on the both servers. Therefore, the individual servers CPU utilization may increase dramatically.

Remember that the servers are not only completing their original workload, but they are duplicating the workload from the other server. A cluster of two servers would be running at under 50% capacity while a cluster of six, on average, would only have to absorb one-fifth of the failed server workload. You could run these servers at 80% capacity in non-failure cases and still have the spare capacity needed for a failure situation. Memory can become a bottleneck in a Domino environment, especially when users perform more advanced functions.

When deciding how to distribute application databases, you have to consider client network traffic and server performance versus client open sessions. If you create too many replicas, you can impact Domino server performance by reducing disk space and overloading the Cluster Replicator tasks. Take the time to understand database characteristics such as the kind of databases (mail or application), the amount of paging, disk queue length, and so on. This will enable you to make the correct decision when estimating the hardware required to provide a particular level of service. You should remember to allow enough disk space for the replicas from other servers in the cluster as well.

Sizing using Work Load Estimator (see Chapter 3, "Sizing Domino for iSeries using the Workload Estimator" on page 19) will greatly assist you in correctly estimating the spare capacity you need to allow for to make clustering function as you would want it to. As a rule on iSeries servers, we recommend sizing the entire configuration as though it would all be fully clustered. This will give you enough additional resource and capacity to successfully implement a significant clustered Domino server.

10.2.8 Domino clustering limitations and recommendations

The limitations of Domino clustering include:

- ▶ A server can participate in only one cluster at a time.
- ▶ All servers in the cluster should share the same set of networking protocols.
- ▶ All servers in a cluster must be in the same domain.
- ▶ You must use hierarchical naming, not flat naming, to implement clustering.

- ▶ WAN clustering is not recommended unless you can guarantee WAN speeds similar to your LAN speeds.
- ▶ Consistent replication formulas for selective replicas across cluster-member databases that share the same selective replication formula must share the same relative path name across clustered servers.

10.2.9 Mail database integration for high availability

One of the most important objectives of clustering is to provide high availability access. Clusters have a big role to play in mail delivery.

Clustering mail.box not recommended

The mail servers themselves can be clustered, but we do *not* recommend clustering the mail.box databases. These mail.box files are specialized Notes databases. They are constantly working and rarely have any content for more than a few milliseconds, which would put a huge and almost constant workload on the Cluster Replicators, seriously slow down the router and the SMTP gateway causing potentially huge performance problems resulting in a massive mail backlog on busy systems. Routed mail and mail for delivery to and at the SMTP gateway should therefore never be clustered.

Mail trapped in mail.box

Mail actually in a mail.box is only there for a very short time, usually. But if a mail item is trapped in a mail.box at a crash, it will have to stay there until the server is finally restarted. If the mail item was not fully posted to mail.box, it will be retried by the sender to the alternative mail.box and despatched. Any incomplete trapped item will be discarded from mail.box when the failed router and server are restarted.

User mail failover

When a user actually clicks the button to send a mail item (that is transfer it from the mail file into a mail.box), and the mail file server has just gone down, the user will still be able to post the mail to mail.box on the next cluster member where their mail file exists as soon as the cluster manager fails them across to it.

Highly available SMTP gateways

We recommend a separate SMTP gateway or multiple gateways. The conversion of inbound MIME and e-mail streams from unknown originating mail systems is *not* a set of tasks you should have running on your live production mail server with several thousand users. This type of mail stream parsing and MIME translation should be done on a dedicated platform or in an iSeries case, a dedicated Domino partition or preferably more than one for resilience.

The SMTP gateway is also not suitable for clustering, but a simple pair of SMTP gateways (Domino partitions) can be made highly available by the use of MX records on the firewall or Internet mail relay, directing inbound traffic normally to one SMTP gateway, and then to the other if the first stops responding or crashes. Alternatively it could do Round Robin if both are up.

Outbound the various mail servers need to be configured with either one SMTP gateway pair or the other as their Domino “least cost routing” next hop. The other SMTP gateway is configured to a slightly higher value. Therefore, the outbound SMTP load can be balanced by the operations staff, with some mail servers pointing “least cost routing” to one SMTP

gateway, and the others to the second one. If either gateway fails, half of the traffic will fail over using the rule set in the "least cost routing" next hop. The other half of the traffic was already there, so it won't be affected. This function provides a simple but effective highly available SMTP gateway configuration.

Mail server failover

The only function that works even after one clustered mail server goes down is mail delivery, as long as the notes.ini variable MailClusterFailover is set to 1. The mail routing from other mail servers then fails over to an alternative (cluster) of the crashed mail server. When you set up mail for your cluster, you need to set up failover for mail routing and mail delivery, directory assistance, and shared mail if necessary.

When a message is about to make its final hop (to the user's home server) and the destination server is unavailable, the router checks if the destination server is a member of a cluster. If the user's home server is still unavailable, the router then delivers the message into the cluster replica of the user's mail file. Again, a cunning combination of alternate routes, least-cost mail routing, and mail delivery failover provides a complete, end-to-end solution for high availability of mail routing, SMTP delivery and despatch, and successful delivery to end users within the group of clustered mail servers. For the domain configuration document, the settings are applied to all servers in the domain.

Passthru servers in a clustered environment

A passthru server is always useful for mobile users when there is more than one server in your Domino environment. This is because the remote clients only use one phone connection to access multiple servers. If use of the passthru server is considered mission critical to provide high availability, you can ensure optimal performance of the server by making it a dedicated passthru server. You can provide high availability for passthru servers by creating a clustered passthru server if you are using local area network connections.

10.2.10 Remote access with iNotes

An alternative that is well worth considering for secure remote access of this type for mobile users is iNotes Web Access. This allows the user to securely retrieve and send mail. They can access their calendar, diary, and address book directly from your corporate secure Web site without the need for any specialized modems or equipment. Or they can access the Web from a customer's premises, their own home, or in an Internet cafe.

10.2.11 Application hub server

The hub-and-spoke replication scheme establishes one central server as the *hub* and other servers as the *spokes*. This replication scheme is recommended because it is generally the most efficient replication topology, which minimizes network traffic, especially in larger organizations. A hub server is a mission-critical server, and setting up a clustered hub server will provide high availability.

10.2.12 Network performance

With Domino clustering, network performance depends a great deal on the network implementation. If you are clustering within a single iSeries box, you will take advantage of loopback, which is described in 10.1.5, "Communication between Domino servers on the same iSeries server" on page 278.

10.2.13 Server_Availability_Index and Server_Availability_Threshold

The availability of cluster servers and the process of load-based failover is based on the comparison of the SERVER_AVAILABILITY_INDEX and SERVER_AVAILABILITY_THRESHOLD.

To display the actual SERVER_AVAILABILITY_INDEX of the server, enter **sh cluster** on the server console. This value is calculated by the following formula:

$$100 - (\text{transactions} / \text{SERVER_TRANSINFO_NORMALIZE})$$

Here the default value for Server_Transinfo_Normalize is 3000.

The transactions are counted within a statistic-interval. The length of the interval may be set by the notes.ini setting:

SERVER_TRANSINFO_UPDATE_INTERVAL=<valueinseconds>

The default here is 15s. The number of intervals that are used to average the transaction counts may be set by SERVER_TRANSINFO_MAX, where the default is 5 (these values usually require no change).

As soon the value of the SERVER_AVAILABILITY_INDEX falls below the SERVER_AVAILABILITY_THRESHOLD (also a notes.ini setting, default is 0), the cluster server becomes *busy*.

Since the SERVER_AVAILABILITY_INDEX is based on transactions only, it does not depend on tasks like full text indexing and agent activity that may consume a noticeable amount of system resources. This means that depending on the type of Notes application used on the server, server tasks running, and available resources, the server's responsiveness may be poor while the transaction count is low as well. Therefore, SERVER_AVAILABILITY_INDEX will not decrease by a lot. In this case, it makes no sense to set SERVER_AVAILABILITY_THRESHOLD to a value of, say 98, since the archivable resolution is poor.

Setting the notes.ini variable SERVER_TRANSINFO_NORMALIZE to a value < 3000 will result in smaller values for SERVER_AVAILABILITY_INDEX for a given transaction count. This allows more appropriate tuning of when the server becomes busy. We recommend that you set the SERVER_AVAILABILITY_THRESHOLD to approximately 50 and adjust the Server_Transinfo_Normalize to meet the requirements.

10.2.14 Tools

The following parameters are available in the notes.ini file for modification and provide additional performance enhancement and monitoring. You can see the full details of these parameters in Table C-2 on page 442.

► RTR_Logging

Enables or disables the monitoring of Cluster Replicator activity (be aware of the considerable overhead on this option, use this only while investigating problems, *never* leave enabled).

► Server_Availability_Threshold

Works directly with the Domino statistic Server.Availability. When Server.Availability reaches the threshold set in the Server_Availability_Threshold parameter, the server begins rejecting user requests. The threshold may need to be set high (95 to 97) for the best failover characteristics.

In addition, `Server_Transinfo_Normalize` may need to be set. This allows you to “normalize” the response times experienced by your Domino server and ensures that the failover processing will occur only when it's supposed to. This value can be tailored for your environment.

Tip: Manually set this parameter to carefully calculate the value.

See 10.2.13, “`Server_Availability_Index` and `Server_Availability_Threshold`” on page 293.

► **Server_MaxUsers**

Specifies the maximum number of active NRPC user sessions allowed on a server.

This is *only* effective for NRPC (Notes Client) users and does not affect HTTP users or sessions. When this number is reached, the server state becomes “MaxUsers,” and the server stops accepting new Database Open requests from NRPC users. This prevents new sessions from starting and stops existing NRPC users from opening another database. They also see the error message:

Server Error: Access to the server is restricted due to maximum number of users.

The default is 0 (unlimited access to server by user). We recommend you use the default.

For further information, see “`Server_MaxUsers`” on page 217.

► **RTR_Cached_Handle_Disable**

Enables or disables the caching of open databases in a cluster:

- 0 - Enables caching of open databases
- 1 - Disables caching of open databases

Cached handles improve response times, but can cause excessive I/O on very busy systems.

► **Server_Cluster_Probe_Timeout**

Default 1 (min) max 120: The Interval the cluster server probe each other for availability. Can cause serious performance issues if set too fast depending on loading.

► **Server_TransInfo_Max**

This is the number of intervals that are used to average the transaction counts. The availability of cluster servers and the process of load-based failover are based on the comparison of the `Server_Availability_Index` and `Server_Availability_Threshold`. The `Server_Availability_Index` is calculated by the following formula:

$100 - (\text{transactions} / \text{Server_Transinfo_Normalize})$

Transactions are counted within the statistic interval `Server_Transinfo_Update_Interval`. The number of intervals that are used to average the transaction counts may be set by `Server_TransInfo_Max`.

The default is 5, which we recommend you do not change.

See 10.2.13, “`Server_Availability_Index` and `Server_Availability_Threshold`” on page 293, for more information.

► **Server_TransInfo_Normalize**

Used when calculating the server availability index to “normalize” the response times observed at the server (that is, it divides the observed response times by this normalize value). Until now, this setting was undocumented, but it is available in both R4.6 and R5.

For the availability index calculation to work properly, the normalize value should be roughly equal to the average Domino transaction time (for the server in question) in milliseconds*100. The default value is 3000 ms, which corresponds to an average response time of 30ms per transaction.

Note: This default setting was appropriate for “the average server” when clustering was first shipped several years ago, but it is too large for the current generation of servers. You should use a lower normalize value with today's faster servers, so loads fail over correctly.

Tip: Be sure to manually set this to a sensible value.

See 10.2.13, “Server_Availability_Index and Server_Availability_Threshold” on page 293, for more information.

► **Server_TransInfo_Update_Interval**

This indicates the transactions counted within the statistic interval. The availability of cluster servers and the process of load-based failover are based on the comparison of the Server_Availability_Index and Server_Availability_Threshold. The Server_Availability_Index is calculated by the following formula:

$$100 - (\text{transactions} / \text{Server_Transinfo_Normalize})$$

Transactions are counted within the statistic interval Server_Transinfo_Update_Interval. The number of intervals that are used to average the transaction counts may be set by Server_TransInfo_Max. The default is 15 seconds, which we recommend that you use and do not change.

See 10.2.13, “Server_Availability_Index and Server_Availability_Threshold” on page 293, for more information.

10.2.15 Server-based tools

This section describes the server-based tools for clustering.

Logging replication

The Cluster Replicator generates Replication Event log records once an hour and records information about all replications performed during that hour. Each Replication Event record provides information about database replications and any currently outstanding errors. The Cluster Replicator generates one Replication Event document for each server with which it replicates. You can force the Cluster Replicator to generate a log record immediately by typing the following command at the server console:

```
TELL CLREPL LOG
```

The log analysis tool

By using the log analysis tool, you can collect all the information related to a specific word that is stored in the log database.

Cluster analysis tools

The Domino server contains a cluster analysis tool to help you determine that clustering has been implemented successfully. If you are experiencing problems with a cluster configuration, running a cluster analysis also provides you with the information you need to troubleshoot problems. The cluster analysis tool tests different aspects of the cluster configuration and reports the results back to the cluster configuration database. Cluster analysis is performed from the administration panel and enables you to specify the types of analysis you want to run.

You set up the cluster analysis tool using a remote console. By running the cluster analysis tool, you test the cluster status for every server in the cluster. The tool can be run on any server in the cluster.

Report types

There are two report types:

► Server

You can run cluster server reports that test:

- **Number of cluster members:** Checks the number of servers in the cluster
- **Consistent domain membership:** Checks that all servers are members of the same domain
- **Consistent protocols:** Checks that servers are running consistent protocols
- **Required server tasks:** Checks that the required cluster tasks are running

► Database

You can run database reports that test:

- **Consistent ACLs:** Checks that access control lists are consistent among replicas
- **Disabled Replication:** Checks databases for disabled cluster replication
- **Consistent replication formulas:** Checks for inconsistent replication formulas among replicas
- **Replicas exist within cluster:** Checks databases for replicas in the cluster

Replication backlogs and Cluster Replicator tuning

During peak activity periods, servers may show an especially high frequency of replication events. Replication backlogs may occur if the Cluster Replicator is unable to handle all replication requests. Complete these steps:

1. Determine which servers are showing replication backlogs by typing the following statement at the server console:

```
show statistic replica
```

2. Examine the Replica.Cluster.WorkQueueDepth statistic in this report.

This statistic shows the current number of modified databases that are awaiting replication. If this value is consistently greater than zero, enabling additional Cluster Replicators may help you to decrease replication backlogs.

Note: You can run more than one Cluster Replicator (CLREPL) at a time to increase performance of the system. However, only one cluster database directory task (CLDBDIR) can be run. The suggested Cluster Replicator per server is the number of servers in the cluster minus one.

10.2.16 Clustering tips

You can designate servers for specific Domino tasks. This can improve system efficiency, especially in large organizations, and make administration easier. Some Domino clustering limitations are listed here:

- ▶ For large, busy clusters, cluster analysis may require several hours or more to complete. You may find it convenient to dedicate an administration client when you run cluster analysis tests for any period of time.
- ▶ Databases tend to grow faster than expected, so it is normally better to over-estimate the demand for the storage space.
- ▶ There are performance impacts when adding multiple Domino servers together into a cluster. These impacts include CPU utilization, increased I/O, and increased network traffic. Putting the server-to-server transactions on a separate network eliminates this impact.
- ▶ As the cluster administrator, you should closely monitor your system for usage patterns during peak periods and adjust cluster resources accordingly.
- ▶ If the server is an application server, where LotusScript or Java applications are heavily used, adding CPUs may be the best solution to get rid of the bottleneck.
- ▶ If processor utilization is constantly close to 100%, consider spreading the server workload within the cluster or adding another processor.
- ▶ Getting started with Domino clustering, you may want to start with just two servers. As your experience and understanding grows, you will want to add more servers.
- ▶ Remember that the servers in a cluster must share the same set of protocols. If a client connects to a server using a protocol, the client must use the same protocol when it fails over on a clustered server. For the iSeries server, this must be TCP/IP. TCP/IP also provides better failover performance than other protocols.
- ▶ Reserve extra memory and CPU cycles for the cluster tasks. The servers in the cluster must be able to take care of the increased workload in case the clients are redirected from another server or servers.
- ▶ Consider disk space very carefully for mail servers. As users start to use more advanced features such as attachments, mail messages typically get larger.
- ▶ You must use hierarchical naming, not flat naming, to implement clustering. Clustering uses the Administration Process (AdminP), which cannot use the flat naming convention.
- ▶ A cluster can contain from two to six servers.
- ▶ Because of the high volume of messages exchanged between clustered servers with replicated databases, we recommend that you dedicate a network to the cluster. This means that you need to define a Notes named network and a port dedicated for the Domino servers in the cluster.
- ▶ A useful guideline for determining the number of Cluster Replicators to run on each server is to set up as many Cluster Replicators as there are cluster members, minus one. For example, if there are four servers in the cluster, enable three CLREPL tasks on each server in the cluster. This way, there will always be at least one CLREPL task available to handle replication requests to another server in the cluster.
- ▶ We recommend that you dedicate a private LAN connection for the cluster's internal traffic.
- ▶ If you have heavily-used databases that consistently overload the Cluster Replicator with update requests, consider running multiple Cluster Replicators. When multiple Cluster Replicators are enabled on a server, they work in parallel to replicate changes to other servers. If one Cluster Replicator is busy replicating changes to one database, a second Cluster Replicator can begin replicating changes to another database. By sharing

replication workloads, multiple Cluster Replicators ensure that data updates are made quickly and stay tightly synchronized.

- ▶ The iSeries server Cluster server makes Domino clusters more attractive in large environments where there may be several thousands of clients.
- ▶ To avoid the distribution of the same document to multiple users' mailboxes, you can also create a document database. Encourage users to create all of the documents to be distributed in this database.
- ▶ Instead of attaching an entire document to a mail message, the user should insert a document link to the document database. The document database, in turn, can be clustered to ensure high availability.
- ▶ When clustering mail databases, two replicas in a cluster should be adequate to guarantee availability at any given time for mailboxes, and still keep the overhead of cluster replication low.
- ▶ Mail database replicas should be distributed roughly evenly among the servers in the cluster to avoid unreasonable growth of workload on one server in the case of failure of another server in the cluster.
- ▶ For application databases that have a lot of changes and updates from the clients or a high transaction rate, do not to create more than three replicas.
- ▶ It is a general recommendation that database replication across the clustered servers be scheduled on an hourly basis to serve as a backup to cluster replication. This would eliminate the possibility of any database updates being lost in the event of hardware or network failure.
- ▶ You can designate servers for specific Domino tasks. This can improve system efficiency, especially in large organizations, and make administration easier.
- ▶ We recommend that you use a passthru server when mobile users need to access a cluster. This way, the client takes advantage of the cluster configuration for the destination servers.
- ▶ When databases are replicated, all I/O operations that complete on the original file are also completed on each replica. When placing database files on a server, place them in an optimal location whenever possible.
- ▶ Replicator memory is a consideration only in large clusters or if the workload is very heavy and several Cluster Replicator tasks need to be run simultaneously.
- ▶ We recommend that a passthru server is used when mobile users need to access a cluster. The client takes advantage of the cluster configuration for the destination servers.
- ▶ The server availability threshold is a key configuration setting for workload balancing. Setting the threshold *too high* can cause user requests to fail unnecessarily. Setting the threshold *too low* can result in poor performance for some users that may have received better service from another server.

To determine the proper value for the server availability threshold, you should start by simply monitoring the server availability index during periods of a normal to heavy load. Once you gather some data on the range of typical values of the server availability index for a server, select an initial value for the server availability threshold. This should be a value toward the lower end of the range of typical values. You should also consider how a server outage may impact server workload. If a server in the cluster fails, the failover capability in Domino clustering will direct clients to other servers in the cluster. To allow for this, you may want to set the server availability threshold to allow “extra” capacity to handle the failover workload.

See 10.2.13, “Server_Availability_Index and Server_Availability_Threshold” on page 293.

- Analysis tests can be extensive on clusters with a large number of databases. If you do not have a dedicated server for analysis tests, you may want to run database report types for problems that you suspect. Or, run the tests during low utilization hours.



Internet and intranet performance tips

More and more applications today are Web-based or have a “Web enabled” part to allow access to data or to share existing application functionality over the Web.

Domino servers can also act as good full integrated Web servers, enabling browser clients to directly access selected databases on the server, to send and retrieve electronic mail, and access newsfeeds for example. The Domino server includes server technology that very quickly and easily transforms Lotus Domino databases and applications into Web applications on the same server. Domino combines the open networking environment of Internet standards and protocols with the powerful application development facilities of Notes.

This enables you to develop a broad range of business applications for the Internet, extranet and intranet. The distinctions between extranet, intranet and Internet are bounded by the physical network connectivity using the TCP/IP protocol.

An *intranet* is an internal organization wide area network (WAN) or Metropolitan Area Network (MAN) that uses the TCP/IP protocol. These intranets are protected from external access (from the Internet) using firewalls and proxy servers.

An *extranet* is any sub-set of your intranet that is deliberately exposed while being made securely available through firewalls to allow business partners and suppliers controlled access to a limited amount of the data and functionality that is normally hidden behind the firewall.

The *Internet* is the “jungle”; it is a publicly accessible wide area network that uses TCP/IP protocol standards to facilitate many forms of communication. A large corporation is usually only exposed to the Internet at their Web site (the company shop window, with real-time trading possibilities as appropriate) and the corporate external e-mail systems.

The firewall and proxy servers provide the intranet and the extranet with the secure TCP/IP communications and protection that are essential to protect the boundary between the Web and e-mail access points and the insecure Internet.

11.1 Internet and intranet protocols

Note: From this point onward, we assume that “intranet” includes extranet.

The Internet and intranet protocols and functions that Domino servers currently support include (Figure 11-1):

- ▶ HyperText Transfer Protocol + HTTP Secure (HTTP/HTTPS) Web serving
- ▶ Simple Mail Transport Protocol (SMTP)
- ▶ Post Office Protocol Version 3 (POP3) and Internet Mail Application Protocol Version 4 (IMAP4)
- ▶ Internet/intranet news feed services (NNTP)
- ▶ Lightweight Directory Access Protocol (LDAP)
- ▶ Web page retrieval (WEB)

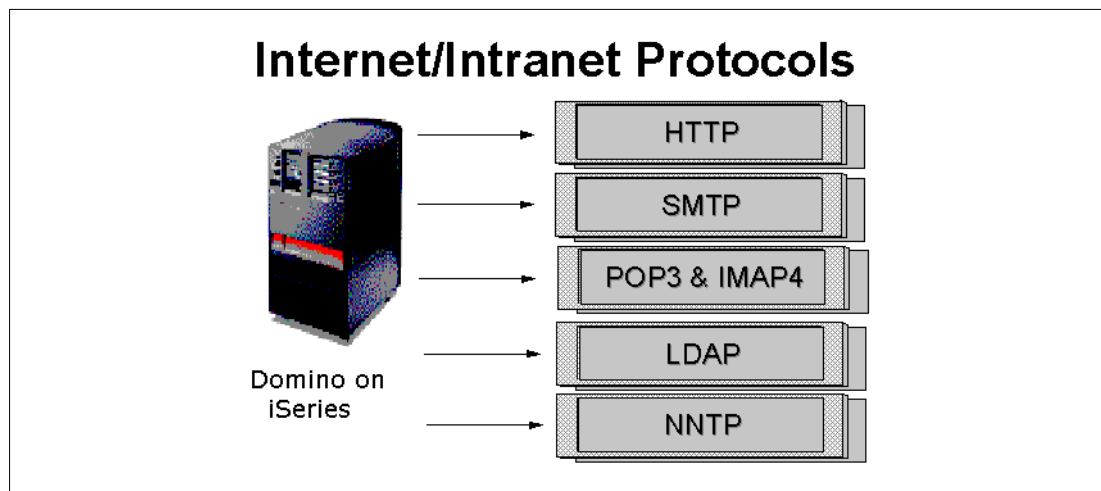


Figure 11-1 Internet protocols for Domino on iSeries

This chapter discusses the performance tuning configurations and considerations of the following base protocols or functions:

- ▶ HTTP performance tuning
- ▶ SMTP performance tuning

In addition, this chapter identifies the impact that the other Internet or intranet protocols can cause to performance on the iSeries server.

The following section assumes that you are familiar with the configuration and setup requirements for the Domino HTTP server.

11.2 HTTP performance tuning

Due to the complex nature of the HTTP protocol, several variables come into play when tuning server performance. Armed with a more in-depth understanding of Domino Web server thread management and its impact on performance, it is possible to tune the Web server for better performance.

Parameters such as “Number active threads” (see “Number of active threads” on page 307) and “Maximum requests over a single connection” (see “Maximum connections over a single connection” on page 307), and “asynchronization” of agent execution (see “Controlling how often Agent Manager runs agents” on page 237) can be used to tune the Domino Web server for maximum performance.

Knowing the impact of these settings allows you to make appropriate changes when tuning your servers. Unfortunately, additional factors strongly affect HTTP performance, namely complexity of application design and user load. It is often necessary for customers to perform their own bench-marking tests to determine optimal configuration. Since so many possible design configurations are possible, it is not realistic for Lotus to provide benchmarking numbers that can suit all customers.

You should consider performance when designing applications. Increased functionality from agents, shared fields, and subforms can lead to a decrease in performance. You should keep in mind that heavily loading up each Domino partitioned server with functionality means it cannot support large numbers of users. Factoring in all of these considerations allows you to optimize thread usage as well other Web server resources (such as memory caches) to achieve the most efficient use of the Domino Web server.

The HTTP server is configured using the “HTTP Server” section within the Domino server document contained in the Domain’s Public Address Book.

The Domino HTTP server is started, stopped, and controlled with some previously existing and several new Domino Console commands. (Additional information can be found in the Domino 5 Administration Help). Table 11-1 shows the basic and new R5 Domino Console commands with a brief description of their actions.

Table 11-1 The new R5 Domino console commands and descriptions.

Domino console command	Description
Load HTTP	Manually start the HTTP server task.
Tell HTTP QUIT	Manually stop the Domino HTTP server task.
Tell HTTP Restart	This command results in the HTTP task shutting down, reloading, and refreshing certain Web server settings. It is the equivalent to issuing “Tell HTTP Quit” followed by “Load HTTP.” However, the down time for the server is much shorter because it is not completely reloading all HTTP memory. This command deletes the in-memory page and user-authentication caches.
Tell HTTP Show Thread State	Gathers information about the status of each worker thread.
Tell HTTP Show Users	Display authenticated users. This command can be used only if the server is configured to use session-based tracking for the Web. This command shows the User Name, IP address, and the time of expiration, which is 30 minutes by default. The results display only users who are authenticated; anonymous users cannot be tracked.
Tell HTTP Show File Access	Displays information about file system protection on the machine and each virtual server if configured.
Tell HTTP Show Security	Displays the current status on the use of SSL for the server and each virtual server or virtual host.
Tell HTTP Show Virtual Servers	Displays a list of all configured virtual servers or virtual hosts running on the server.

11.2.1 HTTP terminology

Some of the terminology regarding the Domino HTTP servers is defined in the following sections.

HTTP server

Hypertext Transfer Protocol (HTTP) is the standard Internet protocol that enables Web clients to talk to Web servers. The Domino Web server “speaks” HTTP and, therefore, enables Web clients to communicate with Domino servers. The server task that allows the Web server to understand HTTP is the HTTP server task that you start when you set up the Web server.

URL interface

The *Uniform Resource Locator (URL)* interface is the standard Internet protocol that enables Web clients to tell Web servers what item they are requesting. The Domino Web server examines the URL in the incoming request and determines if the request is for an item in a Notes database or if it is for an HTML file in the file system.

HTML file request

If the request is for an HTML file, Domino acts like any other Web server and serves the file to the Web client. Domino for iSeries is just as quick doing this with native HTML files as any other Web server.

Notes DB request

When the request is for something in a Notes database, Domino interacts with the Notes database to locate the document and subsequently serve the information to the Web client or conversely to put information from the Web client into a Notes database.

HTML translation

Domino automatically translates Notes features such as navigators, views, documents, and links into HTML for display in the Web client. For example, Notes links and action bar buttons become URLs in the Web client. You simply develop your Web application in Notes, and Domino does the rest. Figure 11-2 shows the main components of Domino regarding HTML translation.

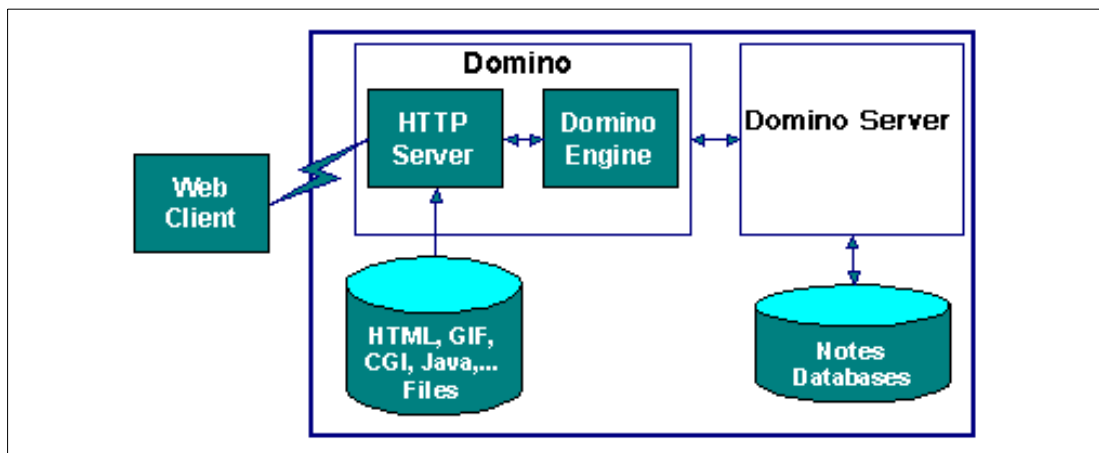


Figure 11-2 HTML translation

11.2.2 Optimizing HTTP threads

The Domino for iSeries HTTP server task is a multi-threaded client/server task. Threads of the HTTP job are prepared and executed for handling incoming HTTP requests from Web browsers.

Unlike Native Notes Client sessions, which are held open for a default period of four hours for each user as “persistent connections”¹, HTTP threads are considered stateless threads. A *stateless thread* is a thread that is not associated directly with a user session. HTTP threads are available for requests to the HTTP server and not associated directly with the number of users currently visiting your Web site.

The number of active threads on a Domino HTTP server represents the number of concurrent HTTP requests that the Web server can handle. The reason why HTTP servers are considered stateless is that the HTTP request must convert them and then send the following information to the Web browser user:

- ▶ Document data
- ▶ Document design
- ▶ Execute all agents prior to conversion (formulas or scripts within the document)
- ▶ Next action information (links, submit, edit, and so on)

This allows the HTTP server to not keep track of previous commands since the Web user has all the required data to execute the next step in navigating the Domino HTTP server. This stateless environment also has a higher overhead than the native Notes Client, since native Notes does not need to pass design and next step information because these functions are built into the Notes Client software.

With Domino R 6, we anticipate significant improvement in the performance of basic file-serving through some major enhancements to the HTTP server task:

- ▶ The threading model has been re-engineered to be much more fair about allocating network resources among clients. This will particularly benefit sites that have heavy file-download traffic.
- ▶ The HTTP task now fully supports HTTP 1.1 “persistent connections,” which should improve performance for users who connect over slow networks because less time is spent establishing and closing connections. This should also impose less overhead on the server constantly re-creating connections.

To specify the number of threads that you want active on your Domino server in Domino R5, use the “Number of active threads” field in the HTTP section of the Server document in the Public Address Book (see Figure 11-3). The default setting is 40 (maximum recommended is 200).

¹ Note, since the introduction of I/O completion ports (IOCP) with Domino for AS/400 5.01.02, each client connection is no longer permanently assigned to a SERVER job. However, it still makes sense to talk about “persistent connections”, because the connection between client and server remains active.

Basics	Security	Ports	Server Tasks	Internet Protocols	MTAs	Miscell
<div> <div>HTTP</div> <div>Domino Web Engine</div> <div>IIOIP</div> <div>LDAP</div> <div>NNTP</div> </div>						
Basics						
Host name(s):						
Bind to host name:		Disabled				
DNS lookup:		Disabled				
Default home page:		default.htm				
Allow HTTP clients to browse databases:		<input checked="" type="radio"/> Yes <input type="radio"/> No				
Maximum requests over a single connection:		1				
Number active threads:		40				
NOTE: The following setting is no longer used in Domino. You should use it only for servers running versions prior to 4.6.						
Minimum active threads:		20				

Figure 11-3 Basics of HTTP configuration for a Domino server

On new installations of Domino R5, the value for “Maximum requests over a single connection” defaults to 1. This means that, by default, the server will not honor the Keep-Alive-Header. As of R5.01, the use of this parameter is disabled, causing the server to ignore persistent connections regardless of this setting.

When the HTTP server task initializes on the Domino server, the defined threads are created and occupy approximately 20 to 40 KB of memory each. These threads are fixed in number until you change the value in the server document. Then, restart the HTTP task.

Use the Work with Active Jobs (WRKACTJOB) command or option 9 of the Work with Domino Servers (WRKDOMSVR) panel and then press F11 twice to confirm that the HTTP job has started the defined amount of threads as shown in Figure 11-4.

```

Work with Active Jobs
ACME01

CPU %:   13.5      Elapsed time:   00:04:37      Active jobs:   240

Type options, press Enter.
  2=Change  3=Hold  4=End  5=Work with  6=Release  7=Display message
  8=Work with spooled files  13=Disconnect ...

Opt Subsystem/Job  User      Number  Type  CPU %  Threads
-----
  DOMIN006      QSYS      514575  SBS    .0      1
  ADMINP        QNOTES    515299  BCI    .0      2
  AMGR          QNOTES    515297  BCI    .0      1
  AMGR          QNOTES    515298  BCI    .0      3
  CALCONN       QNOTES    515301  BCI    .0      1
  HTTP          QNOTES    515570  BCI    .0     48
  QNNINSTS      QNOTES    515291  BCH    .0      1

```

Figure 11-4 WRKACTJOB with 40 plus threads activated for the HTTP job

Measuring thread activity

The following areas regarding threads are discussed in this section:

- ▶ Number of active threads
- ▶ Minimum active threads
- ▶ Idle thread time-out

Number of active threads

The following statistics regarding threads can be measured using the Domino Console command **show stat domino**:

```
COMMAND SENT: show stat domino
Domino.Config.ActiveThreads.Max = 40
Domino.Config.ActiveThreads.Min = 20
Domino.Threads.Active.Peak = 2
Domino.Threads.Peak.Time = 12/13/01 14:32:06
Domino.Threads.Peak.Total = 40
Domino.Threads.Total = 40
```

If `Domino.Threads.Active.Peak` is equal to `Domino.Threads.Total`, HTTP requests may be waiting for the HTTP server to make an active thread idle before handling the request. If this is the case, increase the number of active threads in the HTTP section of the server document (as shown in the above example) until `Domino.Threads.Active.Peak` is less than `Domino.Threads.Total`.

To ensure optimal performance, increase or decrease the number of active threads in jumps of five so that the `Domino.Threads.Total` is close to five threads greater than `Domino.Threads.Active.Peak`. This minimizes the iSeries resources used and provides the greatest user responsiveness.

By having the number of active threads five above the peak active, it also allows for the increased use of your Web site in the future. To ensure that there are sufficient threads in the future, it is important that you review `Domino.Threads.Active.Peak` on a regular basis.

Maximum connections over a single connection

As of R5.01, the use of this parameter is disabled, causing the server to ignore persistent connections regardless of this setting.

Number active threads setting

As mentioned previously, an administrator can set the number of "Number active threads" to allow the server to handle a higher number of concurrent browser connections. However, this number must be optimized appropriately – too low and the resources go unused, too high and there is CPU contention.

Increasing the number of active threads may not actually improve performance. Although increasing threads does not necessarily affect CPU utilization, it increases the use of memory and I/O. In addition, a high number of threads increases context switching, which may lead to slightly higher CPU usage (though not drastic increases).

Determining the optimum number of HTTP threads requires server load tests and tuning. Lotus' current recommendation is to start by setting HTTP active threads to 10% of the estimated number of concurrent Web users. For example, if a customer anticipates 200 mail users on a system, they should begin with 20 active threads. Although Lotus cannot recommend a definite upper limit, we suggest that you do not exceed more than 64 threads per CPU, or half the amount of RAM in MB and generally no higher than 128 threads regardless of the number of processors.

Idle threads time out

From release R4.6.2, including R5 of Domino for iSeries, there are only two statuses for threads:

- ▶ Idle
- ▶ Active

Since the HTTP task in Release 4.6.2 no longer manages the establishment of new threads, the overhead on processing is substantially reduced. For the reasons stated here, we highly recommend that you update your Domino server to release 4.6.2, rather than setting this parameter.

11.2.3 Optimizing HTTP cache settings

There are several cache settings for Domino HTTP servers that include:

- ▶ Images and files
- ▶ Commands
- ▶ Design elements
- ▶ Authenticated users

The following sections describe some configuration settings to influence caching of the Domino HTTP server.

Maximum cache size

Note, the *Maximum cache size setting* is no longer needed with Domino R5 and only valid for previous releases of Domino. If you are using the new release 5 database format (On Disk Structure (ODS) Version 41), you can resave the images and HTML in their native format. Therefore this conversion is no longer necessary. If you have done this step after upgrading to Domino R5, then you will not need to change the cache setting size.

The Maximum cache size is the cache size used by Domino for the HTTP server threads for the conversion process from a Domino database to HTML. (This includes the conversion of text, .gifs, and .jpgs). The default value for this variable is 50 MB. To increase your Domino HTTP server performance, change this to a larger value on Domino servers that are used predominately for HTTP serving. The cache is an amount of storage that is carved out on disk, so it can be set to a rather large value.

The best way to determine an optimal setting is to look at the size of the text and image files that are being served up to see how much space they consume on disk. This will give you a pretty good indication of how large to set this value.

Setting this value is especially important for 4.6.x servers, since prior to 5.0, all text that is served up has to be converted from text into HTML on the fly. This conversion process takes time and CPU. Also, any images that you stored in 4.6.x Domino databases were stored as bitmaps, so they need to be converted back to their native format (.jpgs, .gifs, ...) on the fly as they are served up. The larger you can make this cache size, the more of these converted images and converted HTML can be stored for access by other users.

Maximum cached commands

Caching commands within the Domino HTTP server converts standard Domino server requests to HTTP format and holds the translation within main storage such as:

- ▶ ?OpenDocument
- ▶ ?EditDocument
- ▶ ?OpenView

The design and functionality of your Web site on Domino determines the number of commands required to be cached for optimal performance. To measure the number of cached commands on a Domino HTTP server, issue the following Domino server Console command and note the output:

```
COMMAND SENT: show stat domino
Domino.Cache.Command.Count = 5
Domino.Cache.Command.DisplaceRate = 0
Domino.Cache.Command.HitRate = 0
Domino.Cache.Command.MaxSize = 128
```

If Domino.Cache.Command.Count is equal to Domino.Cache.Command.MaxSize, increase the maximum command cache by five entries at a time until Domino.Cache.Command.Count is less than Domino.Cache.Command.MaxSize. This provides optimized performance for your Domino HTTP server.

Maximum cached designs

The cached design parameter converts Notes database design elements to HTML and stores them in main storage while the server is operational. The number of design elements within your Web site databases determines the number of design cache maximum required to optimize the design cache of the Domino HTTP Web site. This varies depending on the functionality and purpose of your Web site.

To measure the required maximum cached design elements, issue the following command at the Domino Server Console and note the output:

```
COMMAND SENT: show stat domino
Domino.Cache.Design.Count = 20
Domino.Cache.Design.DisplaceRate = 0
Domino.Cache.Design.HitRate = 79.2857142857143
Domino.Cache.Design.MaxSize = 128
```

To optimize the performance of your Domino HTTP design cache, ensure that Domino.Cache.Design.Count is slightly less than Domino.Cache.Design.MaxSize. If these statistics are equal, increase the maximum cached design parameter with the HTTP section of the server document by five until Domino.Cache.Design.Count is less than Domino.Cache.Design.MaxSize. This provides optimal performance for design caching on your Domino HTTP Web site.

Maximum cached users

The *Maximum cached users* and Cached user expiration interval parameter controls the cached information stored in main storage of the OS/400 for HTTP users that have authenticated to the Domino server. Anonymous users are not cached. If your Domino HTTP Web site does not use HTTP user and password authentication, Maximum cached users provides no additional performance improvement.

To measure the required maximum cached users for a Domino HTTP Web server, issue the following command at the Domino server console and note the output:

```
COMMAND SENT: show stat domino
Domino.Cache.User.Count = 1
Domino.Cache.User.DisplaceRate = 0
Domino.Cache.User.HitRate = 87.1794871794872
Domino.Cache.User.MaxSize = 64
```

To optimize the performance of your Domino HTTP user cache, ensure that Domino.Cache.User.Count is slightly less than Domino.Cache.User.MaxSize. If these statistics are equal, increase the maximum cached users parameter with the HTTP section of the server document by five until Domino.Cache.User.Count is less than Domino.Cache.User.MaxSize. This provides optimal performance for user caching on your Domino HTTP Web site.

The cached user expiration period should be left to default (120 seconds). This parameter would be used as a security parameter rather than a performance improvement based on your organization requirements.

11.2.4 Optimizing HTTP logging for performance

The level of logging that can be achieved within a Domino HTTP Web server can be extremely detailed if required. Figure 11-5 shows the logging portion of the HTTP section within a server document in the Domino Directory (formerly Public Name & Address Book or PNAB) that is used to control HTTP logging.

Enable Logging To:		Log File Settings	
Log files:	<input type="text" value="Disabled"/>	Access log format:	<input type="text" value="Common"/>
Domlog.nsf:	<input type="text" value="Disabled"/>	Time format:	<input type="text" value="LocalTime"/>
Log File Names		Exclude From Logging	
Directory for log files:	<input type="text" value="domino\log"/>	URLs:	<input type="text" value=""/>
Access log:	<input type="text" value="access-log"/>	Methods:	<input type="text" value=""/>
Agent log:	<input type="text" value="agent-log"/>	MIME types:	<input type="text" value=""/>
Referer log:	<input type="text" value="referer-log"/>	User agents:	<input type="text" value=""/>
Error log:	<input type="text" value="error-log"/>	Return codes:	<input type="text" value=""/>
CGI error log:	<input type="text" value="cgi-error-log"/>	Hosts and domains:	<input type="text" value=""/>

Figure 11-5 HTTP logging configuration section for HTTP

It is important to minimize the required logging within your Domino HTTP environment since this optimizes the performance of the HTTP server. If logging is not an organizational requirement, it should not be enabled. Logging can be turned on temporarily for troubleshooting if required.

The HTTP logging function is separate from native Domino logging in the LOG.NSF database and requires separate analysis.

11.2.5 Best practices for Domino HTTP server

Consider the following items when looking to provide improved performance for HTTP:

- ▶ Web site design
- ▶ Domino server partitioning
- ▶ Web application categorizing

Web site design

The design of a Domino HTTP Web site is the most critical part to optimize the performance of the Domino HTTP server. All other parameters discussed previously may provide five percent improvement, but a poorly designed site will still perform badly.

The following design features can cause large overheads to the performance of a Domino HTTP Web server and should be taken into consideration:

- ▶ Dynamic @Function agents
- ▶ LotusScript agents
- ▶ Large file attachments such as Java applets
- ▶ Multiple @Functions on a form pre-computed (place them under buttons)

Native HTML pages can sometimes provide serious improvement to Web performance.

Dynamic @Function agents

Dynamic @Functions are used in agents contained in forms, views, and navigators, such as @UserName, @Now, and so on. They cannot be cached since they are considered too dynamic for caching within HTTP access.

For details on the @Functions that cannot be cached, see the document “Expanded command caching in Domino” on the Web at: <http://www.notes.net>

To optimize the performance of your Domino HTTP Web site, minimize the use of formulas of the nature previously described to improve command cache usability.

LotusScript agents

Use of LotusScript within agents contained in forms, views, and navigators cannot be cached for HTTP serving. To improve performance of your Domino HTTP Web server, minimize the use of LotusScript agents within the design of your Web applications where possible.

Large file attachments

If the size of attachments, such as Java applets and images, is minimized, the performance of the Domino HTTP Web site can be dramatically improved since the amount of conversion and image and file cache can be minimized. Concentrate on small, simple images and applets.

Native HTML pages

Domino has two mechanisms for storing native HTML code:

- ▶ Within the body of a Notes document
- ▶ As a pointer to the HTML and CGI directories

Information stored in native HTML requires no conversion process by the Domino HTTP server and can dramatically improve CPU utilization. If the information within your Web site is predominately static, using native HTML code can provide performance improvement.

Domino HTTP server partitioning

The iSeries server platform for Domino provides one of the best partitioned environments since each server partition can be separated from the operating system and other Domino partition memory pools.

The Domino HTTP server can take advantage of this fact by using partitioning of HTTP Web sites to distribute load across multiple partitioned servers running on the same iSeries server. This can substantially increase the total number of HTTP requests that can be submitted to all HTTP servers at any one point in time.

There is also an advantage in separating your Domino HTTP servers from your production Notes servers since HTTP no longer impacts the performance of either system.

Figure 11-6 shows an example of how Domino partitioning on the iSeries server can segregate and improve total performance of Domino HTTP access and native Notes Client access.

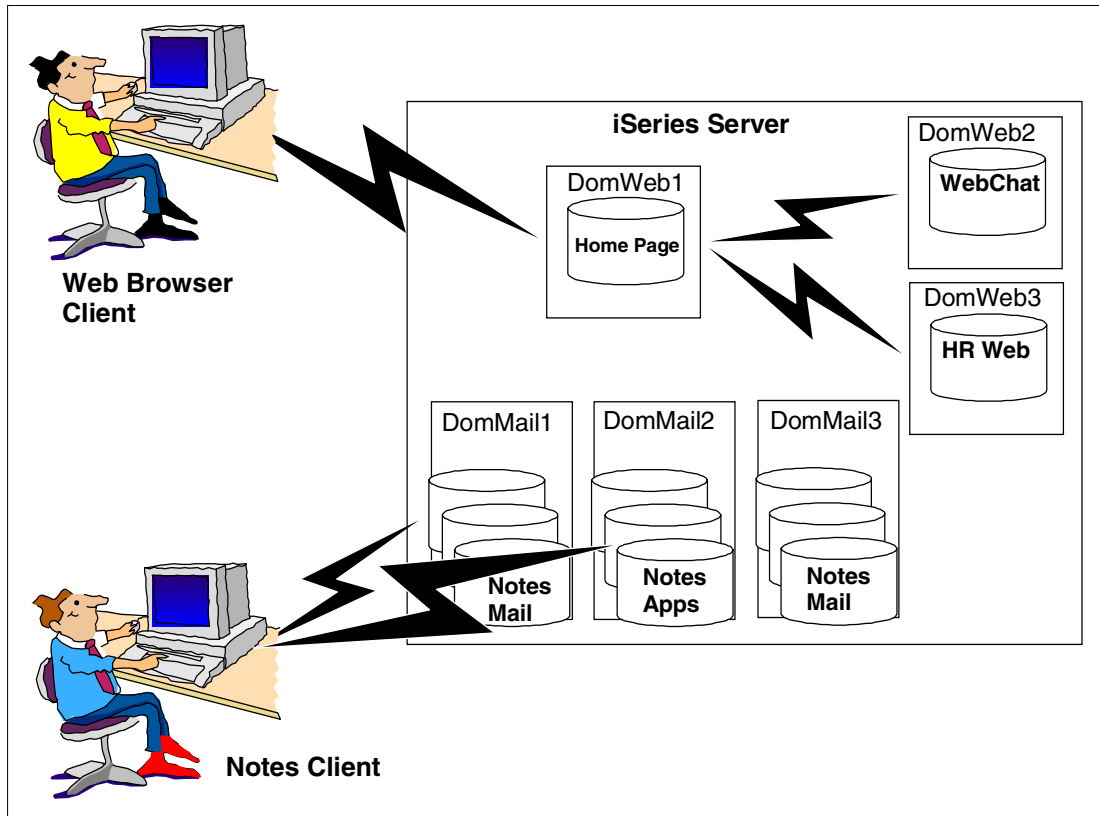


Figure 11-6 Distributing Domino HTTP workload across multiple partitions

This figure shows that the workload for both HTTP and Notes clients can be distributed across multiple subsystems within the iSeries server. Assume that:

- ▶ Each server for Notes Mail (using a Notes client) can sustain 1500 Notes sessions.
- ▶ Each server for the Notes Application (using a Notes client) can sustain 500 Notes sessions.
- ▶ Each HTTP WebChat server (using a browser) can sustain 500 HTTP requests.
- ▶ Each HTTP HR Web server (using a browser) can sustain 250 HTTP requests.

On a single server, we could service (assuming a proportionate mix of each type):

$$\begin{array}{rcccccl} \text{Users totals / server total} & = & 1500/4 & + & 500/4 & + & 500/4 & + & 250/4 & & \\ & & 375 & + & 125 & + & 125 & + & 62 & = & 687 \text{ users (max)} \end{array}$$

On a partitioned server, we could service (assuming we had a sufficient iSeries server):

$$\text{Assume max users on every server} = 1500 + 1500 + 500 + 500 + 250 = 4250 \text{ users}$$

This increases the iSeries server total supported users by 80% (from 687 => 4250).

To achieve the HTTP configuration described here, the home-page application would point to a database stored on another Domino HTTP server to distribute the load, such as the Web application and Web discussion database. This can be simply achieved within the design of the home page application.

Another advantage of splitting the server functions across multiple partitions is that organizational priorities can now be assigned to a particular subsystem job class to ensure that the highest priority tasks (Native Notes Mail & Databases access) of the organization get a larger portion of iSeries server resources than lower organizational priority tasks (such as intranet services). For more information regarding job priorities, see 6.5, “Choosing which processor priority to use” on page 173.

Web application categorization and which HTTP stack to use

Currently three Web servers are available for OS/400:

- ▶ IBM HTTP Server for iSeries licensed program is free of charge and gives you the ability to create two different types of Web servers:
 - The HTTP Server (original) server: The “native” OS/400 HTTP stack also supports Domino databases and WebSphere Application Server through plug-ins.
 - The HTTP Server (powered by Apache) HTTP stack: A port of the Apache Software Foundation's Apache HTTP Server Version 2.0; also has full WebSphere support, but cannot server Domino contents.
- ▶ The Domino HTTP server

Performance advantages of using the OS/400 native HTTP stack

Is there any performance advantage by using the OS/400 “Native HTTP stack plug-in” for serving on the iSeries server compared with Domino's own HTTP server? Many times, we heard the opinion that the Domino HTTP server performance not as efficient was the native HTTP server does. In reality there is no difference, *if* both serve the same type of documents.

In other words, if HTML documents are served, Domino and the OS/400 HTTP server need the same resources and will show the same response times. Much more CPU power is needed, however, if Domino documents need to be converted to HTML. That also applies if you use the Domino HTTP or the Domino plug-in for the OS/400 HTTP server.

Performance would not be a reason to decide to use the OS/400 HTTP stack with the Domino plug-in, because there would be no worthwhile gain. But be aware that from the Domino side, there are some very significant other issues.

Restrictions on using iSeries Domino Plug-in for OS/400 HTTP Server

The Domino plug-in for the OS/400 HTTP Server is new for Domino 5.0.4. The setup instructions are contained in the Domino for AS/400 5.0.4 Release Notes (Readas4.NSF or Readas4.PDF from the Notes.Net Doc Library at <http://www.notes.net/notesua.nsf/>). There are a number of prerequisites and restrictions that should be clearly understood before you attempt to use this feature.

- ▶ Prerequisites:
 - Review information in the Domino for AS/400 5.0.4 Release Notes.
 - OS/400 must be at V4R4 or later.
 - Install 5722-DG1 (or 5769-DG1 for OS/400 Version 4) with the latest Group PTFs²: SF99156 for V5R1 and SF99036 for V4R5.
 - Apply PTF SA89753 to 5769-LNT.
- ▶ Major (Domino) restrictions:
 - Client authentication using SSL is not supported.
 - Domino Administration via the WebAdmin application (WebAdmin.NSF) is not supported.

² See URL <http://ibm.com/servers/eserver/iseries/support/supporthome.nsf/document/10000031> for a list of all iSeries group PTFs.

- Automatic Domino restart after crash not supported. It must be restarted manually.
- OS/400 HTTP Server must be ended before ending the associated Domino server.
- After a Domino server crash, the associated OS/400 HTTP Server must be ended before attempting to restart the Domino server.
- Applications that generate URLs that reference a Domino databases by its unique ID (UNID) instead of its file name (for example, my_db.nsf) are not supported. Domino.Doc is an example of an application that generates URLs in this non-supported format.
- The Domino Plug-in for OS/400 HTTP Server is not supported with QuickPlace.
- Applications that use Domino Web Server API (DSAPI) filters are not supported, for example, Domino Off-Line Services (DOLS) a key part of iNotes Web functionality.
- The Domino Internet Cluster Manager (ICM) is not supported.

Where Domino HTTP fits in best

Domino HTTP servers are best used as dynamic and interactive Web applications due to the native advanced function sets offered within Domino. Static Web serving, even using Domino, is always most efficiently performed by using Native HTML documents at a local file system level since this saves all the overhead of Domino authenticating and then retrieving the data from within a secure Notes database, converting it to HTML, and finally serving it.

Flat documents and pictures in the local file system requiring no HTML conversion will perform much better. Taking this into consideration, we should categorize Web applications that may fit better with Domino HTTP and those perhaps better served by the WebSphere Application server using Native OS/400 HTTP.

The categories of Web applications can be identified as:

- ▶ Static Web application (no change to pages)
- ▶ Informational only (minimal changes to pages)
- ▶ Submission-based Web applications (simple data feedback forms)
- ▶ Interactive Web applications (complex data feedback forms)
- ▶ Workflow-driven Web applications (workflow with SSL or e-commerce)

The native OS/400 HTTP server with WebSphere would be the best option for the static and informational-only Web applications since the dynamic Web application abilities of Domino are not required and here WebSphere will be significantly faster. The disadvantage is that, unlike Domino, which is “Web Application out of the box”, WebSphere will take more setup and programming. This is especially true if you then want the workflow and feedback from the Web to Domino without setting up entirely new data stores.

The Domino HTTP server would be better used for the submission-based, interactive, and workflow Web applications. These take full advantage of the advanced function sets of the Domino HTTP server. With the Domino stack, you also benefit from full SSL authentication to Domino, support for DOLS, iNotes Web and Access, Cluster Manager, and easy restartability.

The decision about whether to use WebSphere, Domino, or both on iSeries needs to be carefully thought out. In future revisions, WebSphere, Domino, and Apache will continue to pull closer together until we standardize on one very flexible and extremely secure, industry standard HTTP stack.

Defining your HTTP applications

As you can see, the possible divisions of the Domino HTTP applications can be very involved. It is important that you set organizational Service Level Agreements for each type of application and divide the functionality according to your organizational requirements. HTTP applications, like Notes applications, can have a huge impact on overall performance.

11.2.6 Summary of HTTP issues, architecture, and tuning possibilities

HTTP is a stateless protocol. This means that HTTP does not allow for a dedicated session or connection between the browser and the server (as is the case with Notes RPC). Instead, when a browser submits an HTTP request for data (typically over port 80), it has a limited connection. In other words, a specific HTTP thread services that socket connection for a limited number of HTTP requests. Once the browser hits this limit, the server forces the browser to make a new connection. The server thread is then freed up to service other pending requests.

In HTTP 1.0, during a single transaction (or socket connection), the server allows the browser to submit only one request, for example, HTML source. Once the browser retrieves the item (HTML file, image, etc.), the connection is closed and the browser is forced to establish another socket connection.

The HTTP 1.1 specification allows for persistent connections. The browser is able to submit additional requests without the overhead of making a new connection. This is an advantage since creating connections can be very time consuming. However, even though connections take time to establish, keeping a connection open also takes resources from the server. Many HTTP servers have a limit on the number of connections they can keep open at one time. Likewise, Domino limits the number of items that can be requested over an open connection before it terminates the connection, freeing up resources.

This limit is configured in the HTTP section of the Server document in the Domino Directory (previously Public Name and Address Book, NAB), in the *Maximum number of requests over a single connection* field. For Domino 4.6x, this value defaults to five requests. This means that any browser submitting requests over a connection can submit no more than five requests before the connection is closed and the browser must request a new connection. Since R5.0.1, we recommend that you change this setting to allow only one request per connection (setting of 1). This is to avoid certain time-out problems with browsers.

The HTTP specification does not limit the browser from submitting several requests over the connection at once, referred to as “pipelining”. However, as of today no browsers currently pipeline requests in a single connection. In other words, the browser waits for the first request submitted to be returned before it submits an additional request (for example, in the case of images). This in turn affects how HTTP threads on the server interact with HTTP sockets.

11.2.7 Performance tuning of time-out values on the Domino Web server

The following sections give a brief overview on what Domino Web server process connection thread is, how it times out, and how time-outs can be tuned for better performance or to avoid connection failures from Web browsers.

How threads work on the Domino Web server

When a browser makes a request to a Domino server HTTP process, the following occurs.

1. A thread is opened.
2. When the request is fulfilled, the thread becomes idle.

3. If the same browser makes a request again, the thread is reused for that browser.
4. If all threads are in use, the server will buffer the transaction request until a thread is free.

Table 11-2 shows the three time-out values that control HTTP time-outs in R5.

Table 11-2 Time-out settings used

Time-out setting	Description
Idle Thread Timeout	This is the length of time that the Domino server should keep an idle thread open. The default is 0, which means the server never drops idle threads. (This was R4 only.)
Input Timeout	This is the time that a client has to send a request after connecting to the Domino server.
Output Timeout	This is the maximum time that the Domino server has to send output to a client.
CGI Timeout	This is the maximum time that a CGI program, started by the Domino server, has to finish.

Figure 11-7 shows the default timeouts for HTTP that may need altering. The Timeouts section is located in the Domino server configuration document under the Internet Protocols tab, HTTP subtab.

Timeouts	
Input timeout:	2 minutes
Output timeout:	20 minutes
CGI timeout:	5 minutes
Idle thread timeout:	0 minutes

Figure 11-7 HTTP default timeouts

Guidelines for tuning HTTP time-out values

When tuning HTTP time-out values, consider these guidelines:

- ▶ **Input timeout** (default 2 minutes): A common error over modem connections, typically seen when users submit a form after having it open for editing for an extended period of time. Increase (as below) by 20% and monitor carefully.
- ▶ **Output timeout** (default 20 minutes): most often observed over slow modem connections when downloading or requesting a large files. We recommend you increase this to 300 minutes.
- ▶ **CGI timeout**: Ensure you know how long it takes your CGI scripts to execute before you set the CGI time-out value. If there are a lot of external processes to run, it may take longer than the default of 5 minutes.
- ▶ **Idle Thread Timeout** (default 0 minutes “no timeout”): Zero means an idle thread will be kept open indefinitely. Setting this to 5 or 10 minutes may improve performance on slower servers or in high-traffic situations. This is ignored in R5. *Applies to Domino R4 only.*

11.2.8 Full Text Search information in a document over the Web

A document is enabled with Full Text Search against a database from the Web and collects various fields from a user. Then it performs a Full Text Search in the database based on the user's input. This works as expected for simple and medium requests. However, complicated searches can return an Error 500 to the user. The search document works without any problems from a Notes client and only fails on a Web client (browser).

This error occurs if the time it takes for the server to return the search results surpasses the time indicated in the (Idle Thread Timeout if R4), Input Timeout, Output Timeout, or CGI Timeout options listed in the HTTP Server section of the Server document.

To resolve this, increase the time-outs listed as Idle Thread Timeout (R4 only), Input Timeout, Output Timeout, and CGI Timeout options in the HTTP Server section of the Server document. The amount of the increase is dependent on the application. However, we suggest that you try increasing the amount by 20%.

11.3 Structure of the SMTP MTA

The SMTP MTA is based on the Internet Request for Comments (RFCs), summarized in Table 11-3. This lists the majority of the RFCs, but should not be taken as a complete and definitive list.

Table 11-3 Key RFCs implemented with Domino R5

RFC	Description
RFC821	Simple Mail Transfer Protocol
RFC822	Standard for the Format of ARPA Internet Text Messages
RFC974	Mail Routing and the Domain System
RFC1047	DUPLICATE MESSAGES AND SMTP
RFC1123	Requirements for Internet Hosts - Application and Support
RFC1495	Mapping between X.400 and 822 messages. (SoftSwitch only)
RFC1496	Downgrading rules between 88 and 84 X.400 (Softswitch only)
RFC1494	Equivalences between X.400 and 822 message bodies (Softswitch only)
RFC1652	SMTP Service Extension for 8bit-MIMEtransport
RFC1700	Internet Assigned Numbers
RFC1734	POP3 Authentication command
RFC1740	MIME Encapsulation of Macintosh files - MacMIME
RFC1741	MIME Content Type for BinHex Encoded Files
RFC1777	Lightweight Directory Access Protocol
RFC1778	The String Representation of Standard Attribute Syntaxes
RFC1823	The LDAP Application Program Interface
RFC1847	Multipart/Signed and Multipart/Encrypted
RFC1869	SMTP Service Extensionsv

RFC	Description
RFC1870	SMTP Service Extension for Message Size Declaration
RFC1891	SMTP Service Extension for Delivery Status Notifications
RFC1892	Multipart/Report Content Type for the Reporting of Mail System Admin Messages
RFC1893	Enhanced Mail System Status Codes
RFC1894	An Extensible Message Format for Delivery Status Notifications
RFC1939	Post Office Protocol (POP version 3) STD 53
RFC1960	A String Representation of LDAP Search Filters
RFC1959	An LDAP URL Format
RFC1985	SMTP Service Extension for Remote Message Queue Starting
RFC 2017	Definition of URL Message/External-Body Access Type
RFC2045	MIME Part 1: Format of Internet Message Bodies
RFC2046	MIME Part 2: Media Types
RFC2047	MIME Part 3: Message Header Extensions for Non-ASCII Text
RFC 2048	MIME Part 4: Registration Procedures
RFC2049	Conform to (MIME) Part Five: Conformance Criteria and Examples
RFC2060	Internet Message Access Protocol - Version 4rev1
RFC2061	IMAP4 COMPATIBILITY WITH IMAP2BIS
RFC 2086	IMAP4 ACL extension (implemented in R5 client)
RFC 2088	IMAP4 non-synchronizing literals (implemented in R5 client)
RFC 2095	IMAP/POP AUTHorize Extension for Simple Challenge/Response (implemented in R5 client)
RFC 2110	MIME E-mail Encapsulation of Aggregate Documents, such as HTML (MHTML) (implemented in R5) Obsoleted by RFC 2557 (see below)
RFC 2111	Content-ID and Message-ID Uniform Resource Locators (obsoleted by RFC 2392)
RFC 2112	MIME Multipart/Related Content-type (obsoletes 1872) (implemented in R5)
RFC 2177	IMAP4 IDLE command (implemented in R5)
RFC 2183	MIME Content Disposition Header Support
RFC 2193	IMAP Mailbox Referrals (implemented in R5 client)
RFC 2221	IMAP Login Referrals (implemented in R5 client)
RFC 2197	SMTP Command Pipelining
RFC 2222	Simple Authentication and Security Layer (SASL)
RFC 2311	S/MIME Version 2 Message Specification
RFC 2312	S/MIME Version 2 Certificate Handling
RFC 2313	PKCS #1: RSA Encryption Version 1.5

RFC	Description
RFC 2314	PKCS #10: Certification Request Syntax Version 1.5
RFC 2315	PKCS #7: Cryptographic Message Syntax Version 1.5
RFC 2557	MIME Encapsulation of Aggregate Documents, such as HTML (MHTML)
RFC 2421	Voice Profile for Internet Messaging (version 2)
RFC 2480	Gateways and MIME Security Multiparts (in 5.01)
RFC 2487	SMTP Service Extension for Secure SMTP over TLS (in 5.01)
RFC 2505	Anti-Spam Recommendations for SMTP MTAs (Best Practices RFC 2/99) (in 5.01)
RFC 2554	SMTP Service Extension for Authentication (in 5.01)

11.3.1 SMTP performance improvements in Domino R5

Release 5 provides a native SMTP router that eliminates the conversion that is typically performed in 4.6x (MIME<-->CD record). This alone significantly boosts overall message routing throughput compared to R4.

R5 MIME goes native

The Notes R5 client implements MIME natively. This means that IMAP and POP3 performance improves over 4.6x as a result of leveraging a new “on-disk” Notes MIME type. Beginning in Domino 5.0, the Notes Mail Router (the Router task) now has the ability to route mail via Notes Remote Procedure Calls (NRPC) or via Simple Mail Transfer Protocol (SMTP).

SMTP for POP3 and IMAP4

Other Internet mail clients that use Domino as a message store, such as POP3 and IMAP4 clients, also require this SMTP support for transferring messages even into the local Notes Domain for other Notes mail users.

Retry interval and limit

Specifying the “retry limit” or “retry interval” is not available on any particular documents. Options are available (see “SMTP notes.ini variables” on page 320) that somewhat control the amount of time the Router will try to route a particular message. By default, the Router will attempt delivery of pending mail for a 24-hour period of time before generating a Non-Delivery Notification report and notifying the sender.

R5 Router includes the SMTP (MTA)

The R5.x Router works differently than the R4 SMTP MTA did, in regard to setting a predetermined number of attempts and the time between these retries. The Router works based off the “Initial Transfer Retry Interval” field on the Server Configuration document in the Domino Directory (NAMES.NSF). The value of this field will be used to determine when a failed message is retried for the first time. The additional attempts to send the message will be based on two times and three times the value of this field. All remaining attempts, after the third retry, will be done at this interval (three times the Initial Transfer Retry Interval) for the total of the 24-hour period.

Example of SMTP timeout and retry parameter

The following example is based on the default value of 15 minutes to illustrate the point:

1. The initial retry is attempted at 15 minutes. Since this is a retry, it is already the second attempt.
2. If the initial retry attempt is unsuccessful, the Router backs off the attempts, by doubling the “Initial Transfer Retry Interval” (here 15 minutes) before trying again.
3. A retry is attempted at 30 minutes. This is now the third attempt to send the message.
4. If the second retry (third attempt) is unsuccessful, the Router backs off the attempt again. This time it triples “Initial Transfer Retry Interval” (here $15 \times 3 = 45$ minutes) before trying again.
5. A retry is attempted in 45 minutes. This is now the fourth attempt to send the message.
6. All remaining attempts are done at the 45 minute interval for 24 hours.

Lowering the value of the “Initial Transfer Retry Interval” increases the retry attempts per hour and could possibly increase the success rate of routing the messages.

Increasing the value of the “Initial Transfer Retry Interval” decreases the retry attempts per hour and results in longer routing times.

Note: The only way to reset the retry interval is to end and start (“recycle”) the router process. Issuing a “route servername” at the console attempts an immediate transfer. But if it is unsuccessful, the next retry interval is used for the next attempt.

SMTP notes.ini variables

Of course, there are the inevitable notes.ini variables that can be set that will increase or decrease when the Router generates a Non-Delivery Notification.

Important: Implementing either of the MailTimeout=counter or MailTimeoutMinutes=mins parameters will apply to Notes-to-Notes mail, as well as Notes-to-SMTP mail.

MailTimeout

The MailTimeout parameter (MailTimeout=<days>) specifies the number of days after which the server returns undelivered mail to the sender. Increase this setting when you have a lot of mail returned in one day or when you are sending mail to foreign domains. To specify a timeout period of less than one day, use the notes.ini MailTimeoutMinutes setting.

There is no default. Although if this setting is omitted, undelivered mail is returned after one day.

MailTimeoutMinutes

The MailTimeoutMinutes parameter (MailTimeoutMinutes=<minutes>) specifies the number of minutes after which the server returns undelivered mail to the sender. The maximum number of minutes is 1440 (24 hours). Increase this setting when you have many messages returned in one day or when you are sending mail to foreign domains. To specify a timeout greater than a day, use the notes.ini MailTimeout setting.

There is no default. Although if this setting is omitted, undelivered mail is returned after one day.

Measuring SMTP performance

You can measure the current performance of the inbound and outbound messages independently as described in the following section and Table 11-4.

Inbound and outbound performance measurements

Statistics on the status of outbound messages can be gathered by issuing the Domino console command. Note the output of the messages:

```
COMMAND SENT: sh stat SMTP.*
SMTP.Command.DATA = 643
SMTP.Command.EHLO = 98
SMTP.Command.HELO = 522
SMTP.Command.MAIL = 643
SMTP.Command.QUIT = 617
SMTP.Command.RCPT = 695
SMTP.Command.RSET = 12
SMTP.MessagesProcessed = 642
SMTP.MsgRecipients.Ave = 1
SMTP.MsgRecipients.Max = 13
SMTP.MsgRecipients.Min = 1
SMTP.MsgRecipients.Total = 694
SMTP.MsgSizeKBytes.Ave = 17
SMTP.MsgSizeKBytes.Max = 2539
SMTP.MsgSizeKBytes.Min = 0
SMTP.MsgSizeKBytes.Total = 11224
SMTP.SessionDuration.Ave = 6
SMTP.SessionDuration.Max = 331
SMTP.SessionDuration.Min = 0
SMTP.Sessions.Accept.Queue = 0
SMTP.Sessions.Active = 0
SMTP.Sessions.Inbound.BytesReceived = 11,068,685
SMTP.Sessions.Inbound.BytesSent = 237,235
SMTP.Sessions.Inbound.non-SSL = 621
SMTP.Sessions.Inbound.Total = 621
SMTP.Sessions.Peak = 6
SMTP.Sessions.Threads.Busy = 0
SMTP.Sessions.Threads.Idle = 6
SMTP.Sessions.Total = 62
```

Table 11-4 shows the MTA, SMTP, and SMTPMTA statistics and their meanings from EVENTS4.NSF from Notes R5.0.8.

Table 11-4 SMTP SMTPMTA and MTA statistics

SMTP MTA statistics	Meaning and value
Mail.TotalRouted.SMTP	Number of mail messages moved from mail.box via SMTP
MTA.Smtp.Dead	Dead messages in SMTP/MIME MTA Inbound and Outbound work queues
MTA.Smtp.TotalKBTransferred	Total number of Kilobytes (Inbound and Outbound)
MTA.Smtp.TotalRouted	Total recipients in SMTP/MIME MTA messages routed (Inbound and Outbound)
MTA.Smtp.TransferFailures	Total SMTP/MIME MTA transmission failures resulting in an NDR or UMN
MTA.Smtp.Transferred	Total SMTP/MIME MTA messages transferred (Inbound and Outbound)
MTA.Smtp.Waiting	Total messages waiting in SMTP/MIME MTA work queues

SMTP MTA statistics	Meaning and value
MTA.Smtp.WaitingRecipients	Number of Responsible Recipients in waiting mail held in Inbound and Outbound work queues
SMTP.<name>.Ave	SMTP average
SMTP.<name>.Count	SMTP count
SMTP.<name>.Max	SMTP maximum
SMTP.<name>.Total	SMTP total
SMTP.Command.<CommandName>	SMTP command
SMTP.MessagesProcessed	Number of SMTP messages processed
SMTP.Sessions.Accept.Queue	SMTP listener work queue of incoming sessions to be processed
SMTP.Sessions.Active	Current number SMTP server tasks
SMTP.Sessions.Inbound.BytesReceived	Total Number of BytesReceived by this SMTP server
SMTP.Sessions.Inbound.BytesSent	Total Number of BytesSent by this SMTP server
SMTP.Sessions.Inbound.Non-SSL	Number of SMTP server inbound TCP connections (non SSL)
SMTP.Sessions.Inbound.SSL	Number of SMTP server inbound SSL connections
SMTP.Sessions.Inbound.SSL.Bad_Handshake	Number of SMTP server SSL Handshake Failures
SMTP.Sessions.Inbound.Total	Total Number of SMTP server inbound connections (SSL and non SSL)
SMTP.Sessions.Peak	Peak number SMTP server tasks since server startup
SMTP.Sessions.Threads.Busy	Number of SMTP server tasks currently running
SMTP.Sessions.Threads.Idle	Number of SMTP server tasks currently idle
SMTP.Sessions.Total	Total number SMTP server tasks since server startup
SMTPMTA.ConvFailures	Total SMTP MTA conversion failures
SMTPMTA.Dead	Dead SMTP MTA messages
SMTPMTA.Delivered	Total SMTP MTA messages delivered
SMTPMTA.HighInBound	Highest number of inbound processes
SMTPMTA.HighOutBound	Highest number of outbound processes
SMTPMTA.InBoundBytes	Total number of bytes inbound
SMTPMTA.InBoundSize	Largest message inbound
SMTPMTA.OutBoundBytes	Total number of bytes inbound
SMTPMTA.OutBoundSize	Largest message outbound

SMTP MTA statistics	Meaning and value
SMTPMTA.TotalRouted	Total SMTP MTA messages routed
SMTPMTA.TransFailures	Total SMTP MTA transmission failures
SMTPMTA.Transferred	Total SMTP MTA messages transferred
SMTPMTA.Waiting	Total SMTP MTA messages waiting
SMTPMTA.WaitingConv	Mail waiting to be converted
SMTPMTA.WaitingRecipients	Mail waiting to be delivered
SMTPMTA.WaitingTrans	Mail waiting to be transferred

11.3.2 Best practices for SMTP messaging using partitioning

Since SMTP messaging is predominately used to connect Domino Mail servers to Internet mail servers, the organizational priority of SMTP messaging would be lower than normal Notes Mail priorities. In addition, in 7.11, “Using a separate partition for the R5 SMTP MTA server” on page 202, we recommend that you run the SMTP MTA in a separate partition so that any MIME conversion or parsing errors, virus detection, and general communications with the mail relays and firewalls will not impact the performance of the Mail servers or the end-user performance or response times. The iSeries server uses partitioning of Domino servers well by placing each Domino server in a separate subsystem on the iSeries server. Each subsystem can be assigned different pool sizes and job run priorities.

To optimize overall Domino server performance for end users, as above, we again recommend that the best practice is to create a separate partitioned server to facilitate SMTP messaging only. Then, manage it independently to the Domino servers accessed by end users.

This SMTP isolation prevents any impact caused by SMTP messaging from impacting end-user access and mail servers. Also the job run priority can be raised for the SMTP MTA server to ensure that end users get first access to CPU resources on the iSeries server.

Taking these points into consideration, you may consider it unimportant to optimize performance for SMTP and allocate the SMTP server the minimal resources that are required to facilitate base messaging. The conversion between the Notes and SMTP message format is also a large overhead on the iSeries server processor and can impact user sessions on that server.

For further details on optimizing the priorities for subsystem jobs, see Chapter 6, “Tuning the iSeries server for Lotus Domino” on page 161.

Separating inbound and outbound SMTP messaging

Partitioning can also be used to create separate SMTP servers for inbound and outbound SMTP messaging. This can distribute the overhead of SMTP messaging across two iSeries server subsystems. This configuration provides high availability and is useful for high-volume SMTP servers. It also provides a significant level of redundancy in the event that one of the SMTP MTA servers becomes unavailable to ensure that SMTP mail still flows in both directions.

Troubleshooting SMTP, IMAP, and POP3 issues for Notes clients

The following parameters are available to troubleshoot POP3 and SMTP issues on the Notes client side of the transaction.

In a Notes 4.6x or 5.x client used as a POP3 client, you can enable a log file for POP3, SMTP, or both protocols. The following steps enable client debugging for POP3 and SMTP.

1. Exit Notes.
2. At the operating system level, create a subdirectory called C:\NOTESLOG.
3. Go into Edit mode in notes.ini, and add the following lines:
SmtplibClientDebug=1
POP3ClientDebug=1
Debug_Outfile=C:\NOTESLOG\DEBUG.LOG
4. Restart Notes.

The following code is a sample for the log after restarting:

```
Sample Output:09:44:47AM
Index
update
process
started
09:44:49 AM POP3Client: GetMessageCount
09:44:49 AM POP3Client: Connect: Host acme1.lotus.com, Port 110
09:44:49 AM POP3 Server: 9.95.49.21 connected
> 09:44:49 AM POP3 Server: [00B0:00A2] Greeting state
09:44:49 AM POP3 Server: [00B0:00A2] AuthenticateUser state
09:44:49 AM POP3Client: ReceiveResponse: +OK Lotus Notes POP3 server version X1.0 ready on
ACME1/MIKEY.
09:44:49 AM POP3Client: Authenticate: User Joe User/Mikey, Pass xxx
09:44:49 AM POP3Client: CommandUSER: Joe User/Mikey
09:44:49 AM POP3 Server: [00B0:00A2] USER command
09:44:49 AM POP3 Server: [00B0:00A2] AuthenticatePass state
09:44:49 AM POP3Client: ReceiveResponse: +OK Joe User/Mikey, your papers please.
09:44:49 AM POP3Client: CommandPASS: xxx
09:44:49 AM POP3 Server: [00B0:00A2] PASS command
09:44:49 AM POP3 Server: JOE USER/MIKEY logged in
09:44:49 AM POP3 Server: [00B0:00A2] Transaction state
09:44:49 AM POP3Client: ReceiveResponse: +OK Joe User/Mikey has 0 messages (0 octets).
09:44:49 AM POP3Client: CommandSTAT:
09:44:49 AM POP3 Server: [00B0:00A2] STAT command
09:44:49 AM POP3 Server: [00B0:00A2] Transaction state
09:44:49 AM POP3Client: ReceiveResponse: +OK 0 0
09:44:49 AM POP3Client: CommandSTAT: 0 0
09:44:49 AM POP3Client: CommandQUIT:
09:44:49 AM POP3 Server: [00B0:00A2] QUIT command
> 09:44:50 AM POP3Client: ReceiveResponse: +OK Lotus Notes POP3 server signing off.
09:44:50 AM POP3 Server: 9.95.49.21 disconnected
09:44:50 AM POP3 Server: JOE USER/MIKEY logged out
09:44:50 AM POP3 Server: [00B0:00A2] Terminal state
> 09:44:50 AM Opened session for JOE USER/MIKEY (Build 147)
> 09:44:52 AM Closed session for JOE USER/MIKEY
Databases accessed: 4 Documents read: 0 Documents written: 0
> 09:45:00 AM Index update process shutdown
02/24/01
09:45:24
AM
Index
update
process
started
09:45:25 AM POP3Client: GetMessageCount
09:45:25 AM POP3Client: Connect: Host acme1.lotus.com, Port 110
```

```

09:45:25 AM POP3 Server: 9.95.49.21 connected
09:45:25 AM POP3 Server: [00B0:00A2] Greeting state
09:45:25 AM POP3 Server: [00B0:00A2] AuthenticateUser state
09:45:25 AM POP3Client: ReceiveResponse: +OK Lotus Notes POP3 server version X1.0 ready on
ACME1/MIKEY.
09:45:25 AM POP3Client: Authenticate: User Joe User/Mikey, Pass xxx
09:45:25 AM POP3Client: CommandUSER: Joe User/Mikey
09:45:25 AM POP3 Server: [00B0:00A2] USER command
09:45:25 AM POP3 Server: [00B0:00A2] AuthenticatePass state
09:45:25 AM POP3Client: ReceiveResponse: +OK Joe User/Mikey, your papers please.
09:45:25 AM POP3Client: CommandPASS: xxx
09:45:25 AM POP3 Server: [00B0:00A2] PASS command
09:45:25 AM POP3 Server: JOE USER/MIKEY logged in
09:45:25 AM POP3 Server: [00B0:00A2] Transaction state
09:45:25 AM POP3Client: ReceiveResponse: +OK Joe User/Mikey has 0 messages (0 octets).
09:45:25 AM POP3Client: CommandSTAT:
09:45:25 AM POP3 Server: [00B0:00A2] STAT command
09:45:25 AM POP3 Server: [00B0:00A2] Transaction state
09:45:25 AM POP3Client: ReceiveResponse: +OK 0 0
09:45:25 AM POP3Client: CommandSTAT: 0 0
09:45:25 AM POP3Client: CommandQUIT:
09:45:25 AM POP3 Server: [00B0:00A2] QUIT command
09:45:25 AM POP3Client: ReceiveResponse: +OK Lotus Notes POP3 server signing off.
09:45:25 AM POP3 Server: 9.95.49.21 disconnected
09:45:25 AM POP3 Server: JOE USER/MIKEY logged out
09:45:25 AM POP3 Server: [00B0:00A2] Terminal state
09:45:25 AM Opened session for JOE USER/MIKEY (Build 147)
> 09:45:26 AM Closed session for JOE USER/MIKEY
Databases accessed: 2 Documents read: 0 Documents written: 0
> 09:45:31 AM Index update process shutdown
09:45:32 AM SMTPMTA: Reloading configuration
09:45:32 AM SMTPMTA: msgcnv Reloading configuration
09:45:32 AM SMTPMTA: osesctl Reloading configuration
09:45:32 AM SMTPMTA: msgcnv Reloading SMTP MTA Configuration
09:45:32 AM SMTPMTA: drt Reloading configuration
> 09:45:33 AM SMTPMTA: isesctl Reloading configuration
> 02/24/01
09:46:03
AM
SMTPClient:
Connect:
Host
acme1.lotus.com,
Port
25,
Connecting
Domain
acme1.lotus.com
09:46:04 AM SMTPMTA: iseshlr0 Startup complete
> 09:46:04 AM SMTPMTA: isesctl iseshlr0 startup completed
09:46:04 AM SMTPMTA: iseshlr0 Attempting SMTP session with
[acme1.lotus.com]
09:46:04 AM SMTPMTA: iseshlr0 Inbound SMTP connection with
[acme1.lotus.com] has been established.
09:46:04 AM SMTPMTA: iseshlr0 Active SMTP session with
[acme1.lotus.com]
09:46:04 AM SMTPClient: ReceiveResponse: 220 ACME1.LOTUS.COM Lotus SMTP MTA Service Ready
09:46:04 AM SMTPClient: CommandHELO: Connecting Domain acme1.lotus.com
09:46:04 AM SMTPClient: ReceiveResponse: 250 ACME1.LOTUS.COM
09:46:04 AM SMTPClient: SubmitMessage:

```

```

09:46:04 AM SMTPClient: CommandMAIL: ReversePath mikey@acme1.lotus.com
09:46:04 AM SMTPClient: ReceiveResponse: 250 OK
09:46:04 AM SMTPClient: CommandRCPT: ForwardPath Joe_User@Lotus.com
09:46:04 AM SMTPClient: ReceiveResponse: 250 OK
09:46:04 AM SMTPClient: CommandDATA:
09:46:04 AM SMTPClient: ReceiveResponse: 354 Enter Mail, end by a line with only '.'
> 09:46:05 AM SMTPMTA: iseshlr0 Inbound message from [acme1.lotus.com] has been
safe-stored.
09:46:05 AM SMTPMTA: iseshlr0 Message notification sent to Inbound Message Converter
> 09:46:05 AM SMTPClient: ReceiveResponse: 250 Message received OK.
09:46:05 AM SMTPClient: Disconnect:
09:46:05 AM SMTPClient: CommandQUIT:
09:46:05 AM SMTPMTA: iseshlr0 Inbound SMTP connection with
[acme1.lotus.com] has been closed.
09:46:05 AM SMTPClient: ReceiveResponse: 221 GoodBye

```

IMAP, LDAP, NNTP, and SMTP IP addressing for partitioned servers

In R4, the notes.ini parameter SMTPMTA_IPADDR=*n* provided an alternative way to use a designated Network Interface Card (NIC) particularly on partitioned Domino servers. The alternative way is to set this variable to the IP address that has been assigned to a NIC. For example, a service provider is running multiple partitioned servers on an iSeries server, and has an SMTPMTA on each partitioned server for each different company. Any SMTP inbound mail to a specific partitioned server on this iSeries is received and processed by the corresponding SMTP MTA of that server, for example:

```
SMTPMTA_IPADDR=11.22.33.44
```

R5 does *not* have this notes.ini parameter. However, by using a combination of two notes.ini variables, we can achieve this functionality.

The R5 “InternetServers” in this context means (IMAP, LDAP, NNTP, SMTP, ICM) support for partitioned servers. This is the *one* area that is not backward compatible with R4.6. The new way to set this up is more in line with the Domino Port Driver Model. Logically an Administrator needs to bind a Notes TCP Port to a specific address and then tell the Internet Server to use that Notes port. The process of binding an IP address to a Notes port has been required since R3 and R4 for Notes Domino servers and is accomplished by the following notes.ini setting:

```
<NotesTcpPortName>_TcpIpAddress=0,IPaddress:1352
```

Port 1352 here is actually optional but refers to the Domino TCP port number.

The Internet Servers need to be told what Notes TCP Port Name to use. If not specified, the Framework chooses the first one it sees. The Notes TCP Port Name used by an Internet Server must be specified by adding the following notes.ini:

```
<InternetServer>NotesPort=<NotesTcpPortName>
```

Here <Internet Server> equals IMAP, LDAP, NNTP, SMTP, or ICM. Each Internet server can only support one Notes TCP port.

Here is an example of the notes.ini file:

```

TCPIP=TCP, 0, 15, 0
TCPIP1=TCP, 0, 15, 0
TCPIP2=TCP, 0, 15, 0
TCPIP3=TCP, 0, 15, 0
TCPIPETH1=TCP, 0, 15, 0
TCPIPETH2=TCP, 0, 15, 0

```



```

TCPIP_TcpIpAddress=0,10.40.180.192:1352
TCPIP1_TcpIpAddress=0,10.40.180.193:1352
TCPIP2_TcpIpAddress=0,10.40.180.194:1352
TCPIP3_TcpIpAddress=0,19.16.14.192:1352
TCPIPETH1_TcpIpAddress=0,10.76.32.92:1352
TCPIPETH2_TcpIpAddress=0,10.76.32.93:1352
$$HasLANPort=1
Ports=TCPIP,TCPIP1,TCPIP2,TCPIP3,TCPIPETH1,TCPIPETH2
SMTPNotesPort=TCPIP
LDAPNotesPort=TCPIP1
NNTPNotesPort=TCPIP2
SMTPNotesPort=TCPIP3
ICMNotesPort=TCPIPETH1

```

First associate an IP address with a Notes port, for example:

```
<YourNotesTcpPortName>_TcpIpAddress=0,<your IP address>
```

Then, configure the Internet servers for which Notes TCP Port name to use:

```
<InternetServer>NotesPort=<NotesTcpPortName>
```

11.4 Other Internet and intranet protocols and functions

The performance optimization of the remaining intranet or Internet protocols and functions would be solely based on your organizational requirements for these functions. There is little additional configuration that can be set to offer specific performance improvement for these tasks.

Where priorities for these protocols can be separated, the Domino servers hosting these protocols should be partitioned to ensure that the impact of one Internet or intranet protocol server does not impact another Domino server.

The following sections describe briefly the functionality of these additional intranet or Internet protocols and functions.

11.4.1 IMAP or POP3 on Domino Server within a cluster and failover

IMAP or POP3 running on a Domino server within a cluster cannot use the Domino failover technique. To have failover functionality, you need the Notes client since this is the only client that will detect unavailability of the server and possible failover to a cluster from the Cluster Manager. IMAP and POP3 clients do not have this functionality.

However, with either IMAP or POP3, you can use a “Round Robin” using TCP/IP and DNS (see 11.4.3, “POP3 server” on page 329) or use a Router that can direct the client to an available server within the cluster.

11.4.2 IMAP server

IMAP clients can download mail to a local mail file or interact with and manage mail directly on a Domino server that runs the IMAP service. They use the IMAP protocol to read and manage mail, use SMTP to send mail, and use LDAP to access the Domino Directory. Lotus Notes Release 5 includes IMAP client functionality.

Additional IMAP client support

IMAP services on the Domino server have been enhanced to provide for wider support of IMAP clients, such as PINE, Eudora, Cyrusoft Mulberry, and Execmail 5.1 clients, as well as Netscape, IE and Notes.

IMAP performance: Imposing session limits

Domino administrators in R5.0.3 now limit the number of sessions that the IMAP server will allow. A notes.ini parameter - IMAPMaxSessions - allows administrators to specify the maximum number of sessions that will be allowed in the IMAP server. If the parameter is not specified, or if the parameter is specified with a value of 0, then no limit will be enforced.

R5 IMAP variations: IMAP Online, IMAP Offline, and IMAP Disconnected

The principal differences between IMAP Online, IMAP Offline, and IMAP Disconnected are documented in Table 11-5.

In 5.0 (R5), both the Notes client and the Domino server are IMAP compliant. This means that all of the following scenarios can use IMAP mail:

- ▶ Notes client with a non-Domino IMAP server
- ▶ Non-Notes IMAP client with a Domino server
- ▶ Notes client with a Domino server

For this last scenario, most people would communicate over NRPC. However, customers who want non-proprietary protocols can have Notes clients communicate over IMAP to a Domino server.

Regardless of the scenario you use, you need to enable the R5 mail file for IMAP by one of the following methods:

- ▶ Specify that the mail type is IMAP during user registration.
- ▶ Load the Convert task with the -m option (for a mail file that was not IMAP-enabled during user registration)

Three types of IMAP available in R5.0: Online, Offline and Disconnected. Each type is described in Table 11-5.

Table 11-5 IMAP types, purposes, and configuration

IMAP type	Purpose	How to configure in a Notes R5.0 client
IMAP Online	<p>To allow an IMAP client to display and manipulate messages and folders stored on an IMAP server. The IMAP user can create, rename, move, and delete folders. The user can also send, edit, copy, paste, and save messages.</p> <p>Analogous to a Notes client or a browser reading a mail file on a Domino server.</p>	<p>Create an Account document in which you specify IMAP as the type and Online as the access method.</p> <p>This creates a proxy database in the client's data directory. The proxy database allows you to see and manipulate documents in the server-based mail file even though the proxy NSF itself does not store any message documents.</p> <p>Close and re-open the Account document and click the Open Proxy button in the Action bar.</p> <p>The proxy database displays the contents of the server-based mail file even though the proxy database stores no documents itself.</p>

IMAP type	Purpose	How to configure in a Notes R5.0 client
IMAP Offline	<p>To download messages stored in a mail file on an IMAP server to the client and then disconnect.</p> <p>Very similar to how a POP3 client downloads messages from a POP3 server with the exception that the IMAP server does not delete messages after transferring them to the client. This supports the concept of Universal Inbox whereby a Notes client can download and store messages from various sources – POP3, IMAP, and Domino – all in the R5 Notes mail file. The mail file can be either local to the client or server-based. The user could also use Notes replication to replicate this Universal Inbox.</p>	<p>Create an Account document specifying IMAP as the type and Offline as the access method.</p> <p>No proxy database is created.</p> <p>In the Location document, specify the mail file that will receive the downloaded messages.</p> <p>Open the replicator and click the Send and receive mail button.</p> <p>The messages on the IMAP server download to the specified mail file.</p>
IMAP Disconnected	<p>The disconnected mode allows for the following functionality:</p> <ul style="list-style-type: none"> ▶ Editing a local mailbox on the IMAP client ▶ Editing a server-based mailbox (Online) ▶ Synchronizing the documents and folders between the two. <p>This synching is analogous to Notes replication. However since the protocol is IMAP instead of NRPC, the following differences exist even if the server and client are both Notes:</p> <ul style="list-style-type: none"> ▶ Design elements do not synchronize. ▶ There are no replication formulas, although you can select a subset of folders to synchronize. 	<p>Follow the configuration steps for IMAP Online.</p> <p>Then make a local replica of the proxy database. The local replica contains a copy of all messages and all folders in the IMAP mailbox. The user can manipulate the local replica while disconnected.</p> <p>To synchronize the local and server mail files, open the replica of the proxy, which is local, and select File-> Replication-> Replicate. The interface is the same as replication but it is really invoking the IMAP protocol and uses the proxy database. The proxy does not have to be open in the GUI for synching to occur.</p> <p>One final point is that Notes R5.0 is one of the first products to implement the disconnected mode, which is part of the IMAP RFC. Netscape has also implemented this.</p>

For a listing of IMAP clients and more information on IMAP, see the IMAP Web site at:
<http://www.imap.org>

11.4.3 POP3 server

The Post Office Protocol Version 3 (POP3) is an Internet mail protocol that allows a client running POP3, such as Netscape Navigator, Eudora Pro, or Microsoft Internet Explorer, to retrieve mail from a host server also running POP3. You can set up a Domino server to be a POP3 host server for POP3 clients. POP3 clients do not run Notes, but have Notes mail files on the POP3 server. They periodically connect to the server to retrieve their mail. To use a Domino server as a POP3 server, run server task POP3. Then, create a mail file on the server and a Person document in the Public Address Book for each POP3 client that will use the server as its POP3 host.

POP3 clients do not use Notes IDs, and their Person documents do not contain public keys. Therefore, they are not authenticated through Notes, and they cannot read encrypted mail. A Notes mail file can be accessed from both Notes and POP3 clients.

Failover and load balancing

Failover and load balancing can be achieved by the configuration of the POP3 client and the DNS or Host files in your environment. First, the DNS server or local Host file must be configured to respond for both servers using one host name. This is achieved by listing the server host name twice with two separate IP addresses. For example, in a local Host file, this would be:

- ▶ Pop3server 1.1.1.ServerA
- ▶ Pop3server 1.1.1.ServerB

This is an example of using Round Robin DNS. This also provides rudimentary rather crude but quite serviceable load balancing between the two POP3 servers. It doesn't actually load balance; it simply alternates every other inbound request equally between the hosts. If a host fails, the re-directed request fails. The IP address will be made "unavailable" for a period (this is a TCP/IP parameter) and then retried again. It will be repeated at usually ever increasing intervals until the failed server is up and running again.

Note: You must ensure that when you are configuring your POP3 client that you use this shared hostname with multiple IP addresses to identify the POP3 server farm, rather than a specific IP address, which is just a single host without "cluster support".

11.4.4 LDAP server

As with other components of the Domino infrastructure, full support for the latest open standards gives customers maximum flexibility. It enables each organization to leverage their Domino Directory infrastructure as its unique needs dictate.

Domino and LDAP V3

Domino R5 server continues to enhance Domino and Notes standards support by implementing the latest version of the LDAP. Lotus provides the most comprehensive LDAP support available anywhere, with a full implementation of the new LDAP Version 3.0. This includes UTF-8 character set support, language tags, full protocol support, SASL/SSL support, Secure Sockets Layer (SSL) V3, retrieval of X.509 User Certificates, LDAP URL referrals, and attributes defined by the Lightweight Internet Person Schema (LIPS), and more.

LDAP makes Domino highly accessible

LDAP support in Domino makes directory information highly accessible. It lets mail clients, applications, and other servers securely access information in the Domino Directory. For example, organizations can make appropriate directory information (such as a white pages) securely available via Web browsers outside the corporate firewall. Domino for iSeries R5 LDAP supports the Notes R4 and R5 Clients, Microsoft IE5 and IE6, Netscape 6.2, and the ever popular NC Navigator V3 for IBM Network Station (5648-B10).

LDAP and X.509 security

Robust security allows Domino customers to specify exactly what directory information is visible to any LDAP client and how searches can be configured. LDAP support also lets Domino servers and applications, and Notes clients, access other directory servers that support LDAP. This enables Domino to act as a proxy server for any Web browser, LDAP client or Notes client. It provides a common interface for access to information in the directories of suppliers and other business partners, for example. Domino R5 continues its

X.509 support of being able to store, create, and authenticate X.509 Version 3 certificates. It maintains our own built-in Notes certificates and in R5 added support for using X.509 Version 3 for certification purposes, providing enhanced interoperability by allowing a single certificate to be used across multiple applications.

Continuation references

Continuation references are a key new area now fully supported above Domino R5.0.5. They allow an organization to distribute a directory tree across multiple directory servers. When an LDAP client specifies a search base when searching an LDAP server that is configured to hold continuation references, the server can return URLs (referrals) that allow the search to continue over on additional servers that potentially hold entries applicable to that search base.

Domino LDAP service cannot return continuation references, only referrals. A referral is an alternate server address (along with that of usually one replica) that a directory server returns when its directory does not contain a search base specified by a client. A referral is a form of error message, while a continuation reference is a positive response indicating that one or more servers have to be traversed to complete a search.

Note: LDAP Directory vendors do not necessarily use the term “continuation references”; for example, Netscape uses the term “Smart Referrals.”

How to view LDAP statistics in Domino R5

Domino R5 creates statistics when the LDAP task runs. You can view these statistics in several different ways. For example, on the server console, when you type **show stats ldap**, the statistics are written to the Log, as shown in the example below.

In addition, if the Event and Collect task are running on the server, or you are collect the statistics via another server, the LDAP stats are written to the STATREP.NSF database on the collecting server. You can see these stats by checking the Document Properties for the document Statistics Report.

However, because the Statistic Report form does not include defined fields for these statistics, you must customize the form and create an additional view in order to see them in a view or in the document. To do this, follow these steps:

1. Open the **Statistics Report** form in the Designer.
2. Create a new section named **LDAP**.
3. Create a table with two columns.
4. For each LDAP statistic, create a field. Use the statistic name for the field name.

Note that, because the following LDAP statistics contain a space or are longer than 32 characters, they cannot be used for field names:

- ▶ LDAP.Anonymous LDAP Connections
- ▶ LDAP.Sessions.Inbound.BytesReceived
- ▶ LDAP.Sessions.Inbound.BytesSent
- ▶ LDAP.Total LDAP Connections
- ▶ LDAP.Total LDAP Search Entries Returned
- ▶ LDAP.Total LDAP Searches

Another way to view the LDAP Stats in the STATREP.NSF is to populate them to a rich text field. If you are using Domino Release 5.0.2 or later, follow these steps:

1. Open **STATREP.NSF** in the Designer.
2. Copy the **Platform Report** form and rename it to LDAP Report.

3. In Globules Declarations, replace “Platform Statistics” with LDAP Statistics Report:
Const PLATFORM_SECTION_TITLE = “LDAP Statistics Report.
4. In the “QueryOpen” LScript Code, replace “platform*” with ldap*:

```

Forall item In doc.items
  If Lcase(item.name) Like "ldap*" Then
    platformStatCount = platformStatCount + 1
    statReport.appendStat item.name +":", item.values
  End If
End Forall

```
5. Copy view 1 (Statistics Reports\Platform) and rename it to 1. Statistics Reports\LDAP.
6. In the Form-Formula, replace “Platform Report” with LDAP Report.
7. Delete all columns except columns 1 and 2.

A Statistic Report from this view shows that all LDAP statistics are now populated to the body of the document (Figure 11-6).

Table 11-6 Sample statistics report generated from LDAP

Session information	Value
Boot ID:	4088521
Statistics collected at:	10:29:26 yesterday
Reporter task running for:	(hr:min:sec)
Server Location:	
Server Administrator:	
LDAP Statistics	
LDAP.Anonymous LDAP Connections:	89
LDAP.Sessions.Accept.Queue:	0
LDAP.Sessions.Active:	0
LDAP.Sessions.Inbound.BytesReceived:	1246
LDAP.Sessions.Inbound.BytesSent:	1246
LDAP.Sessions.Inbound.non-SSL:	89
LDAP.Sessions.Inbound.Total:	89
LDAP.Sessions.Peak:	1
LDAP.Sessions.Threads.Busy:	0
LDAP.Sessions.Threads.Idle:	1
LDAP.Sessions.Total:	89
LDAP.Total LDAP Connections:	89

For further information, refer to the Script Library “StatReportBuilder” in STATREP.NSF.

Extracting the Domino LDAP schema

Application developers need access to the Lotus Domino LDAP Schema to program the correct attributes to search for. For an immediate listing of the latest LDAP schema, Domino itself can provide this information. To extract it, you must use the LDAPSEARCH utility that is included with every installation of the Notes client or Domino server. This is a simple command line utility that can perform LDAP queries against a Domino LDAP server, and can also redirect output for LDIF import, etc.

The command line to obtain the schema is:

```
LDAPSEARCH -h hostname -b "cn=schema" -s base "(objectclass=subschema)"
```

The resulting output is very large and details the entire LDAP schema of the Domino server. It is usually best to redirect this output to a text file.

Another option to obtain the needed information is to publish the schema in a Domino database. To do this, type the following command at the Domino server terminal window:

```
TELL LDAP EXPORTSCHEMA
```

The Domino LDAP Schema (SCHEMA50.NSF) database is then created. This database allows for full text searching on the entire contents of the schema and can be very useful for looking for a particular attribute or object class.

Note: The database does *not* act on the schema in any way; its entire purpose is to list the schema for informational purposes. Also note that, if the schema is extended or modified, changes will not automatically appear in the database; rerun the TELL LDAP EXPORTSCHEMA command again to update it.

LDAP performance

It should be remembered that LDAP is a Notes application, running on the Domino Application Architecture or Framework, the same as any other Notes application. The LDAP data is stored in some highly tuned LDAP notes databases. LDAP is basically a name and address lookup and fast find mechanism, so when tuning LDAP you should make maximum use of the latest ultra fast Full Text Indexing (GTR and EDC) functionality and make use of optional temporary work space options to minimize the impact LDAP has on other Domino partitions, and maximize its speed, keeping the LDAP users happy with the service.

Global text retrieval (GTR)

Domino for iSeries now includes an alternate GTR search engine beginning in Release 5.0.5. The GTR 34 engine is activated by placing the FT_LIBName=ftgtr34 entry in the server's notes.ini file. For details, see 7.6, "GTR search engine version 3.4" on page 199.

Extended Directory Catalog (EDC)

The Extended Directory Catalog is a new and much faster type of directory catalog available in R5.0.5. For additional information on this feature and further details on setting it up, see the Release Note, particularly the "Extended Directory Catalog" section in the "Things you need to know" section of the Release Notes. Some early deployment work at IBM Research with an IBM Worldwide Extended DirCat (EDC) indicates a substantial (approximately 20%) CPU reduction that would also improve overall server performance (including the router). See 7.7.2, "Using Extended Directory Catalog to improve performance" on page 200.

View_Rebuild_Dir

The View_Rebuild_Dir parameter (View_Rebuild_Dir=<path to desired directory>) allows you to change the temporary work directory where the UPDATE task keeps all of its work files. It also causes all views, including (after R5.0.3) permuted (categorized) views, to be rebuilt using the faster R5 Optimized view rebuild. The Optimized view rebuild uses the system TEMP directory (or the specified directory) to generate all temporary files (.DTF files) instead of the R4 default of the DATA directory. We recommend that for LDAP on iSeries, you place this directory outside the data directory path, preferably on its own ASP. See "View_Rebuild_Dir" on page 222.

Useful LDAP Web sites

For more information, search these sites on the Web:

- ▶ LDAP v3 Proposal/Draft:
<http://www.critical-angle.com/ldapworld/draft-ietf-asid-ldapv3-protocol-04.txt>
- ▶ LDAP v2 RFC 1777: <http://andrew2.andrew.cmu.edu/rfc/rfc1777.html>
- ▶ U of Michigan LDAP site: <http://www.umich.edu/%7Eersug/ldap/ldap.html>
- ▶ SSL v3 and LDAP v3:
<http://www.critical-angle.com/ldapworld/draft-ietf-tls-ssl-version3-00.txt>

11.4.5 NNTP server

When you run the Network News Transport Protocol (NNTP) on your Domino server, the server becomes a Domino NNTP server. You enable NNTP in one of three ways:

- ▶ Selecting an NNTP server during the Domino server setup
- ▶ Adding NNTP to the ServerTasks setting in notes.ini
- ▶ Loading the NNTP server tasks after your Domino server is running

A Domino NNTP server lets users participate in private discussion groups or public USENET newsgroups using a Notes Designer client, a standard NNTP newsreader (such as Netscape Communicator Collabra or Internet Explorer news), or a Web browser. Each newsgroup is stored in a separate database.

Table 11-7 shows the actual console commands and the resulting actions.

Table 11-7 NNTP Newsgroup console commands and actions

Command	Actions
Tell NNTP newgroup groupname	Creates a new newsgroup. Use this command to create newsgroups that are not automatically created during a newsfeed.
Tell NNTP Newgroup groupname pathname	Tells NNTP to add a group to its current cache list with the specified groupname and pathname. This is used when a group is created via the template.
Tell NNTP Quit	Stops the NNTP task.
Tell NNTP Show Config	Displays the NNTP server configuration settings specified in the NNTP section of the Server document.
Tell NNTP Show Groups	Displays the names and path names of newsgroups on the server.

Table 11-8 shows the complete list of available statistics for monitoring NNTP.

Table 11-8 NNTP statistics and descriptions

R5 statistic name	Statistic functional description
NNTP.<RemoteServerName>.Articles.Posted	Number of news articles posted by the server "RemoteServerName".
NNTP.<RemoteServerName>.Articles.Sent	Number of news articles sent to the server "RemoteServerName".
NNTP.<RemoteServerName>.Bytes.Received	Number of bytes received from the server "RemoteServerName".
NNTP.<RemoteServerName>.Bytes.Sent	Number of bytes sent to the server "RemoteServerName" by the NNTP.
NNTP.<RemoteServerName>.Pull. Articles.Failed	Number of failed news article transfers from the server "RemoteServerName".
NNTP.<RemoteServerName>.Pull. Articles.Offered	Number of news articles offered by the server "RemoteServerName".
NNTP.<RemoteServerName>.Pull. Articles.Requested	Number of news articles requested from server "RemoteServerName".
NNTP.<RemoteServerName>.Pull. Articles.Transferred	Number of news articles transferred from "RemoteServerName" to the NNTP.
NNTP.<RemoteServerName>.Push. Articles.Failed	Number of failed news article transfers to server "RemoteServerName".
NNTP.<RemoteServerName>.Push. Articles.Offered	Number of news articles offered to the server "RemoteServerName".
NNTP.<RemoteServerName>.Push. Articles.Requested	Number of news articles requested by the server "RemoteServerName".
NNTP.<RemoteServerName>.Push. Articles.Transferred	Number of news articles transferred to the server RemoteServerName by this server.
NNTP.Articles.Posted	Number of news articles posted to the NNTP server.
NNTP.Articles.Sent	Number of news articles sent from the NNTP server.
NNTP.Bytes.Received	Number of bytes received by NNTP server.
NNTP.Bytes.Sent	Number of bytes sent from NNTP server.
NNTP.Pull.Articles.Failed	Number of failed news article transfers received by the NNTP server
NNTP.Pull.Articles.Offered	Number of news articles offered by the NNTP server during pull feeds.
NNTP.Pull.Articles.Requested	Number of news articles requested from the NNTP server during pull feeds.
NNTP.Pull.Articles.Transferred	Number of news articles transferred by the NNTP server during pull feeds.
NNTP.Push.Articles.Failed	Number of failed news article transfers by the NNTP server during push feeds.

R5 statistic name	Statistic functional description
NNTP.Push.Articles.Offered	Number of news articles offered by the NNTP server during push feeds.
NNTP.Push.Articles.Requested	Number of news articles requested from the NNTP server during push feeds.
NNTP.Push.Articles.Transferred	Number of news articles transferred by the NNTP server during push feeds.
NNTP.Sessions.Accept.Queue	NNTP listener work queue of incoming sessions to be processed.
NNTP.Sessions.Active	Current number NNTP server tasks.
NNTP.Sessions.Inbound.BytesReceived	Total Number of BytesReceived by this NNTP server.
NNTP.Sessions.Inbound.BytesSent	Total Number of BytesSent by this NNTP server.
NNTP.Sessions.Inbound.Non-SSL	Number of NNTP server inbound TCP connections (non SSL).
NNTP.Sessions.Inbound.SSL	Number of NNTP server inbound SSL connections.
NNTP.Sessions.Inbound.SSL.Bad_Handshake	Number of NNTP server SSL Handshake Failures.
NNTP.Sessions.Inbound.Total	Total Number of NNTP server inbound connections (SSL and non-SSL).
NNTP.Sessions.Outgoing.non-SSL	Total Outgoing non-SSL TCP Connections.
NNTP.Sessions.Outgoing.SSL	Total Outgoing SSL TCP Connections.
NNTP.Sessions.Peak	Peak number NNTP server tasks since server startup.
NNTP.Sessions.Threads.Busy	Number of NNTP server tasks currently running.
NNTP.Sessions.Threads.Idle	Number of NNTP server tasks currently idle.
NNTP.Sessions.Total	Total number NNTP server tasks since server startup.

NNTP status code interpretation

Each response from an NNTP server starts with a numeric status code. The codes indicate the response to the last command the server received from the client. Status response lines begin with a three-digit numeric code.

The first digit of the response broadly indicates the success, failure, or progress of the previous command:

- 1xx** Informative message
- 2xx** Command successful
- 3xx** Command successful so far, send the rest of it
- 4xx** Command was correct, but couldn't be performed for some reason
- 5xx** Command unimplemented, or incorrect, or a serious program error occurred

The next digit in the code indicates the function response category:

- x0x** Connection, setup, and miscellaneous messages
- x1x** Newsgroup selection
- x2x** Article selection
- x3x** Distribution functions
- x4x** Posting
- x8x** Nonstandard (private implementation) extensions
- x9x** Debugging output

For specific error codes and their meanings, refer to RFC 977 - Network News Transfer Protocol at the following sources:

- ▶ <http://sunsite.auc.dk/RFC/rfc.cgi?number=977>
- ▶ <http://www.internic.net/rfc/rfc977.txt>

NNTP performance enhancement

NNTP replication performance was improved in R5.0.3 to spend less time pulling articles. The nntpcl5.ntf template is required to take advantage of the performance improvements. If you use NNTP, make sure you have performed a Design Refresh on the all your NNTP data bases with the latest version of the template nntpcl5.ntf.

Check the NNTP server configuration settings specified in the NNTP section of the Server document. Make sure NewsGroups doesn't become your servers prime time activity, unless you are a News Service, of course!

11.4.6 iNotes

iNotes is a client brand that was announced at Lotusphere Orlando 2000. It is one of the three Lotus client brands along with Notes and Mobile Notes and consists of three separate products:

- ▶ Domino Off-Line Services (DOLS)
- ▶ iNotes Access for Microsoft Outlook
- ▶ iNotes Web Access

iNotes represents general client access (primarily browsers) to Domino. It is licensed under the Client Access License (CAL) program.

iNotes Access for Microsoft Outlook

iNotes Access for Microsoft Outlook is part of the iNotes client offering that provides Microsoft Outlook 98 and 2000 the ability to use Domino server as a mail and calendaring backend via MAPI. It installs a set of DLLs that provides additional Microsoft private MAPI functions to the existing MAPI services developed for Notes.

iNotes Access for Outlook uses many services of Notes such as replication and local security without the requirement of a full Notes client. iNotes Access for Microsoft Outlook also gives the Outlook user better mobile access via Domino Offline Services and again better server platform coverage and all the other places where Domino outshines when put in comparison to Exchange.

Citrix Metaframe is not a supported platform for iNotes Access for Microsoft Outlook.

Performance of iNotes Access for Microsoft Outlook

The iNotes DLLs make Outlook look just like a Notes client – using NRPC as the access protocol. As such, Domino servers are scaled exactly the same way as they are for Notes clients. Also there is no additional software or services required or installed on the Domino server to enable access from an Microsoft Outlook client.

Clustering

Because iNotes Access for MS Outlook is identical (to the server) to a Notes Client, it has all the advantages of a true Notes client including Clustering and failover.

iNotes Web Access

iNotes Web Access is the next generation of browser-based access to Domino messaging and Calendaring & Scheduling (C&S) and Personal Information Manager (PIM) functionality. iNotes Web Access is based on Dynamic HTML (DHTML), which is a combination of HTML, XML, DOM 2, and Java Script and offers feature-rich browser-based access to Domino mail and C&S. It initially shipped on Win 32 (98, 2000, NT) and will eventually replace the current Webmail offering once platform parity is reached.

Because iNotes Web Access requires only a Web browser, users have easy access to their Notes mail file from just about any location: Internet Cafe, Kiosk, or another person's desk. This trait also makes iNotes Web Access ideal for a shared PC scenario in which multiple users access their messaging and PIM from a single machine.

iNotes Web Access was designed specifically to provide messaging and PIM functionality. Therefore, although very powerful in its implementation, the product provides a very straightforward and intuitive interface.

From the administrator's perspective, iNotes Web Access is an ideal client for simple, cost-effective deployment and maintenance. Being a thin client with a server-based deployment model and minimal to no training requirements, iNotes Web Access will enable administrators to get their users up and running in no time.

Refer to the document titled "What is iNotes Web Access?" (#182718) and <http://www.lotus.com/inotes> for more information on iNotes Web Access.

The only supported browsers that can be used to access iNotes Web Access mail files is Win32 Internet Explorer 5.01 Service Pack 1 or later. See "iNotes Web Access 1.0 Release Notes" (#186566). (Support for IE6 is anticipated but not guaranteed in R5.0.10.)

Unsupported browsers include:

- ▶ Internet Explorer 5.0 Macintosh Edition
- ▶ Pocket Internet Explorer for IPAQ 3670 running Windows CE 3.0
- ▶ Netscape Navigator

iNotes Web Access encryption and signing

The first release of iNotes Web Access does not support signed and/or encrypted e-mail. This feature is being considered for a future release of iNotes Web Access. Encrypted mail viewed through iNotes Web Access displays a message stating that the message is encrypted and cannot be displayed in a browser. It then instructs the user to reopen the message using a Notes client.

iNotes performance

iNotes Web Access is a straight Domino Web application. It has some extremely complicated HTML, XML, DOM 2 and JavaScript coding, but it is a Web Application and must be sized as such. Compared to our previous offering in this class (Notes Web Mail), the expected user ratio compared to Notes clients is much higher, possibly as high as 4:1 (one iNotes Web Access equals four Notes clients).

Obviously with so little processing happening at the client end (ignoring DOLS for the moment), all significant processing of Mail and C&S function must now occur at the server end, therefore, the ratio of users. And as with any Notes Application and Web Application, the performance is based on what the users *really* do, be they power users or humble mail readers.

End-user workstation requirements

Despite what we said above, note that iNotes Web Access is much richer in function than the previously available *WebMail* solution. Therefore, there is much more logic running on the Web browser. Based on customer feedback, we recommend that you have at least a Pentium II 500 MHz with 128 MB memory (256 MB for Windows 2000 or newer) to have a reasonable performance and response time on client side.

See also the Lotus whitepaper *iNotes Web Access Deployment Guide for Domino 5.0.9*, which you can download from the Web at: <http://www.notes.net>

You should also refer to the redbook *iNotes Web Access*, SG24-6553.

Domino Off-Line Services (DOLS)

Domino Off-Line Services was the first component of the iNotes brand to ship, with the first release in first quarter of 2000. DOLS consists of a server component, a DSAPI extension to the Web service, and a client component – the iNotes Sync Manager. DOLS gives browser users the ability to take Domino Web-based applications offline via the browser, work with the application offline, and synchronize the local application with the server-based copy of the application. From a conceptual standpoint, it is the same idea as using the Notes client to create a local replica and replicate changes between the local and server-based database.

Refer to the document “Domino Off-Line Services Frequently Asked Questions” (#179258) and <http://www.lotus.com/dols> for more information on DOLS.



The iSeries Dedicated Server for Domino

This chapter introduces you to and provides details on the *Dedicated Server for Domino*. The iSeries Dedicated Server for Domino is the member of a family of servers built especially for running Lotus Domino R5, unique in the industry. The Dedicated Server for Domino provides the reliability, manageability, and availability on which the iSeries has built its reputation. These systems are packaged as low-cost-to-own, low-cost-to-buy boxes designed specifically for Domino applications.

Lotus Domino offers the leading software for collaboration, messaging, and Web-enabled productivity that spreads quickly in an organization. What starts as casual e-mail in one department rapidly expands to become a critical messaging infrastructure. A simple form application can grow to workflow, then human resources self-service, sales-force automation, and customer care on the Web. Soon, Domino is at the heart of your business, helping everyone work smarter and more effectively.

With the iSeries Dedicated Server for Domino, you can add new Domino applications without adding new server footprints in your organization. The unique subsystem architecture of the iSeries combined with the partitioning capabilities of Domino enables a single iSeries Dedicated Server for Domino to support multiple Domino servers at the same time. Domino servers are reliable and easy to manage from your home office or remotely from the other side of the world through familiar graphical interfaces.

12.1 Dedicated Server for Domino

First announced in January of 1998, Lotus Domino for iSeries has achieved tremendous success in the marketplace. It offers the rock-solid reliability that customers seek as their e-mail, collaborative, and Web-enabled applications become mission-critical. Through ease of management and the ability to reliably run a mixed workload on a single server footprint, AS/400 and iSeries servers deliver low total cost of ownership as Domino servers. IBM and Lotus offer iSeries customers two server choices to meet their Domino needs:

- ▶ The traditional AS/400 and iSeries servers support both Domino work and other workloads, such as line-of-business ERP and on-line transaction processing applications.
- ▶ The Dedicated Server for Domino is designed specifically to support Domino work, either in a stand-alone environment or as a frontend to other types of work running on separate servers. With the latest V5R1 announcements, we've expanded the definition of "Domino work" as you'll learn in 12.2.1, "Enhancements with OS/400 V5R1" on page 343.

In April 2001, IBM and Lotus announced the third generation of Dedicated Server for Domino processors, which build on Model 270 and 820 technology. The five new processor features deliver 20% greater CPU capacity and two times the memory growth compared to their predecessors. By focusing the processing power on Domino, the Dedicated Server for Domino delivers a significantly better Domino price/performance than other iSeries servers. Even the cover of the Dedicated Server for Domino, with its yellow accent panel, emphasizes its design – processing capability targeted for Domino applications.

The April 2001 announcement also included Version 5 Release 1 of OS/400, which provides significant Domino performance improvements with respect to Java and Web processing capability. Availability for V5R1 and the new Dedicated Server for Domino processor features occurred in May 2001. A resave version of OS/400 V5R1, available via PRPQ 5799-DSD after September 28, 2001, provides expanded Dedicated Server for Domino capacity for workloads complementary to Domino such as Java servlets and WebSphere Application Server.

The importance of Domino as a component in e-business implementations, together with Java and WebSphere, continues to grow. This enhanced V5R1 support, available for all Dedicated Server for Domino models including the Model 170, makes the power and price/performance of Dedicated Server for Domino available to customers who want to take advantage of Domino's strength as an e-business application server.

Note: In October 2000, IBM announced the IBM @server iSeries servers. All Model 270 and 820 Dedicated Server for Domino servers are part of the iSeries family. We use the term *iSeries* throughout this chapter to discuss the Model 270 and 820 Dedicated Server for Domino servers.

12.2 Dedicated Server performance behavior and capacity

Prior to the V5R1 Dedicated Server for Domino enhancements, the simplest implementation for the Dedicated Server was a "pure Domino" implementation, such as e-mail and applications that use "out of the box" Domino templates and capabilities. However, many customers and business partners want to integrate Domino on the Dedicated Server for Domino with other applications and data. Now, with the enhanced support in OS/400 V5R1, functions such as Java servlets and WebSphere Application Server are considered *complementary workloads* when they are used in conjunction with Domino.

In conjunction means that Domino-based processing *must* be present on the server. Complementary workloads have access to the full capability of the Dedicated Server, with limitations for the amount of DB2 related work, however. The DB2 database components of both complementary and Domino-based workloads on the Dedicated Server for Domino are allowed to use up to a total of 15% of the system CPU. These concepts are developed in more detail below.

Note: When determining whether a Dedicated Server is the right iSeries server for deploying applications that will be considered complementary on the Dedicated Server for Domino, consider the question:

“Does it make sense to run these applications independent of Domino?”

If the answer to this question is “yes”, and it is likely that the applications will require more than 15% of the CPU being consumed, then you should consider an iSeries server without a non-Domino CPW limit such as an iSeries server with a base processor. This will ensure that you have constant access to the full performance capability of the server even when Domino processing is not present.

12.2.1 Enhancements with OS/400 V5R1

IBM and Lotus launched the original Dedicated Server for Domino in 1999 in response to the marketplace. Customers and business partners wanted the characteristics of OS/400 as a Domino server (reliability, scalability, ease of management) without paying for the ability to run non-Domino applications such as ERP. The outstanding Domino price-performance of the Dedicated Server for Domino is the response to this request.

Since then, the role of Domino in e-business solutions has grown dramatically. Domino is often a critical e-business component, in partnership with other e-business components such as WebSphere Application Server and Java servlets. Customers and business partners have occasionally reported difficulty in determining whether the Dedicated Server for Domino will deliver its price-performance in these environments, given our original definition of *Domino work*.

The marketplace has spoken and we have responded again. With the enhanced V5R1 Dedicated Server for Domino support that became available in September 2001, we are greatly expanding support for complementary workloads (which we describe in detail below). Users can be confident that applications will run well on the Dedicated Server for Domino when they are used in conjunction with Domino-based processing (also described in detail below).

Such processing as file serving, virus detection software for Domino, Java agents and servlets, and WebSphere Application Server can all run as complementary processing and have access to the full performance capability of the Dedicated Server. These V5R1 advantages are supported on all Dedicated Server for Domino models going back to the initial AS/400 Model 170 Dedicated Server offering.

Dedicated Server performance behavior with V5R1

Dedicated Server for Domino performance behavior *with the V5R1 Dedicated Server for Domino enhancements* can be summarized as follows:

- 100% of the CPU is available for Domino-based applications.

Note: Domino-based applications are invoked by Domino function, but the function being called may be a non-Domino function.

- ▶ 100% of the CPU is available for *complementary*, non-Domino workloads in support of Domino applications.
- ▶ 15% of the CPU is available to DB2 database processing.
- ▶ 15% of the CPU is available for all processing when no Domino applications are in use.
- ▶ 0% of the CPU is available for 5250 interactive processing, with an allowance for a single 5250 interactive user to perform all necessary system administration activity.

For administration activity, we recommend that you use Operations Navigator, since it does not need 5250 interactive processing.

Dedicated Server performance behavior before V5R1

Dedicated Server for Domino performance behavior *prior to the V5R1 Dedicated Server for Domino enhancements* can be summarized as follows:

- ▶ 100% of the CPU is available for pure Domino applications.
 - Pure Domino applications have all processing contained within the Domino function.
 - Pure Domino is more restrictive than Domino-based.
- ▶ 15% of the CPU is available for non-Domino applications and DB2 database processing.

All processing that is not pure Domino is subject to the 15% CPU guideline, including DB2 database processing.
- ▶ 0% of the CPU is available for 5250 interactive processing, with an allowance for a single 5250 interactive user to perform all necessary system administration activity.

For information on Dedicated Server for Domino behavior prior to the V5R1 Dedicated Server for Domino enhancements, see the white paper “Evaluating Appropriate Workloads for the IBM eServer iSeries 400 Dedicated Server for Domino”, which you can find on the Dedicated Server home page at: <http://www.iseries.ibm.com/domino/dsd.htm>

Dedicated Server for Domino V5R1 enhancement installation

Since the Dedicated Server for Domino enhancement became available after the general availability of OS/400 V5R1, it may or may not be installed on your iSeries Dedicated Server for Domino even if you have V5R1 installed. For all Dedicated Server for Domino servers that have been shipped before September 2001, a refreshed version of the iSeries Licensed Internal Code (Level RSC) and the latest Cumulative PTF package (C1254510) needs to be installed.

PTF RE01200 is an identifier that indicates whether your system already has the RSC level of the iSeries Licensed Internal Code installed. Use the following command to determine whether PTF RE01200 is already on your system:

```
DSPPTF LICPGM(5722999) SELECT(RE01200)
```

Installing the Dedicated Server for Domino enhancements

If PTF RE01200 is not on your system, you do not have the RSC level installed. If you want to install this new code, complete the following steps:

1. Install the Licensed Internal Code.
 - a. Use the enclosed Licensed Internal Code CD as your installation media in your Dedicated Server for Domino.
 - b. If your system is up and running, put it into restricted state.
 - c. Using manual installation, perform task 1 of the instructions in Chapter 4 of the manual *iSeries Software Installation*, SC41-5120.

- d. At the end of task 1, select Option 1 on the IPL or Install the System menu. This option performs an Initial Program Load (IPL) of the operating system.
2. Install the latest Cumulative PTF package. Use the instructions included in this package to install the latest Cumulative PTF package (C1254510).
3. If you installed any individual Licensed Internal Code PTFs before this installation process, you need to install those PTFs again.
4. If your Dedicated Server for Domino is a Model 270 Processor Feature 2452, you should obtain and install Hiper PTF MF27217. This PTF could provide an additional performance improvement.

For reference purposes, this set of materials is packaged as PRPQ 5799-DSD.

12.2.2 Dedicated Server for Domino workload environments

Workload environments for the Dedicated Server with the V5R1 Dedicated Server for Domino enhancements fall into the categories: Domino-based, complementary, and not appropriate. This section explores each of these categories in more detail.

Domino-based applications

Domino-based applications are applications that are started by a Domino, Sametime, or QuickPlace server. You can observe the jobs and threads of Domino-based applications on an iSeries or AS/400 server using Operations Navigator or the WRKDOMSVR or WRKLQPSVR commands and option 9.

The Domino-based category includes more applications than what we described in previous Dedicated Server for Domino white papers as “Appropriate for the DSD”. Beginning with the enhanced support in V5R1, Domino applications that make external program calls and use functions, such as Java servlets, now have full access to Dedicated Server for Domino capacity, provided their use of DB2 database processing is below the 15% recommended guideline. These enhancements enable the Dedicated Server for Domino to reach its full potential supporting Domino applications that serve as a frontend to legacy data on another iSeries or AS/400 server.

You can expect Domino-based applications to take full advantage of the price/performance characteristics of the Dedicated Server. Examples of Domino-based applications include:

- ▶ Domino e-mail
- ▶ Domino applications that use Java agents and servlets
- ▶ Domino applications or agents using only Domino databases
- ▶ Domino.Doc
- ▶ QuickPlace
- ▶ Sametime
- ▶ Domino Workflow
- ▶ Domino applications that use normal integration methods (for example @db, LS:DO, DECS, LEI, LC LSX) to access DB2 databases on *another* iSeries or AS/400 server

Note: These integration methods take advantage of the OS/400 *Distributed Relational Database Architecture (DRDA)* support.

DRDA support is considered DB2 database processing but generally falls well within the 15% recommended DB2 database processing guideline. Information about potential exception cases is provided in 12.3.3, “Accessing external databases on a different iSeries server” on page 355.

We expect that other Domino *add-in* applications besides Domino.Doc will fall within the Domino-based category. Domino add-ins that come with Domino and customer-written add-ins that make frequent use of Domino APIs are also considered Domino-based. We do not have complete test results for other add-in applications. Their omission from this list should not be interpreted as evidence that they will not perform well. See a related discussion in 12.3.4, “Java and Domino integration” on page 356.

Complementary processing

This new category of processing has been introduced in response to the need for the Dedicated Server for Domino to allow for enhanced integration with functions such as Java servlets and WebSphere Application Server. The intent for complementary processing is that it be used with or in direct support of Domino, where Domino is the primary server workload.

Complementary processing has access to the full CPU capacity of the Dedicated Server for Domino. Processing that is essentially non-Domino in nature is considered complementary when it is used in conjunction with Domino-based processing. Again, *in conjunction with Domino-based processing* means that Domino-based processing must be present on the system. All DB2 database processing must be within the 15% CPU guideline. Examples of complementary processing include Java servlets, WebSphere Application Server, virus detection software for Domino, and other applications and workloads such as file serving with Integrated xSeries Server.

Inappropriate processing

Dedicated Server for Domino servers are designed to support Domino work, they are not an all encompassing server. As a result, there are workloads that do not fit the Dedicated Server for Domino system.

The following applications and processing are not appropriate for the Dedicated Server for Domino:

- ▶ Interactive 5250 applications
- ▶ Applications that exceed the DB2 database 15% CPU processing guideline
- ▶ Applications that are not used in conjunction with or in support of a Domino-based application

Table 12-1 provides a summary of the main types of applications that are inappropriate for the Dedicated Server for Domino along with a description of the resulting performance behavior when you run these applications on the Dedicated Server for Domino.

Table 12-1 Workloads not intended for the Dedicated Server for Domino servers

Application description/limitations	Resulting performance behavior when limitation is exceeded
Interactive 5250 applications: Limited to approximately 0% CPU, with an exception for a single 5250 session to perform all necessary systems administration functions.	Multiple interactive jobs attempting to use CPU resource will experience increased response times. Other processing on the system will not be impacted.
All DB2 database processing, regardless of what type of applications are generating the DB2 database processing, must use less than 15% CPU.	If the 15% CPU guideline is exceeded, the specific jobs or threads using DB2 database resources may experience increased response times. Other processing on the system will not be impacted.

Application description/limitations	Resulting performance behavior when limitation is exceeded
<p>Applications that are not Domino-based will be limited to using 15% CPU when not run in conjunction with a Domino-based application. We also refer to this kind of processing as “non-Domino”. Non-Domino processing is limited to 15% CPU and includes DB2 database processing. If these applications are used in conjunction with Domino-based processing, they will be considered complementary and have access to the full Dedicated Server for Domino CPU capacity. If your Domino application invokes an RPG or COBOL program using LS:DO, the execution time spent in the RPG or COBOL program is considered complementary.</p>	<p>Because this situation by definition occurs when no Domino-based processing is present, Domino processing is not affected. Any non-Domino processing that exceeds the 15% guideline might experience increased response times. In addition, you might observe increased system overhead.</p>

Several remedies are available for a workload that is not appropriate for a Dedicated Server configuration:

- ▶ Make sure non-Domino processing is used in conjunction with Domino-based processing, thereby allowing the non-Domino processing to be treated as complementary.
- ▶ If just slightly more than 15% CPU is required for DB2 database processing on a current Dedicated Server for Domino and no additional DB2 database processing is anticipated, upgrade to a larger Dedicated Server with a higher processor CPW rating.
- ▶ If significantly more than 15% CPU is required for DB2 database processing, you should implement it on a traditional iSeries server.
- ▶ If interactive 5250 processing is regularly required for more than system administrative processing, you should implement on a traditional iSeries server with a larger interactive CPW.

Keep in mind that, in general, traditional iSeries servers with standard or base processors are the best choice to provide real-time integration between Domino applications and DB2 on the same server. This chapter explores the behavior and performance of light DB2 data integration on the same server to help with your planning and analysis. However, our bottom line recommendation for data integration is either to use a Dedicated Server as a frontend to another iSeries or AS/400 server or to run Domino applications on a suitably configured traditional server that houses your DB2 data.

12.2.3 Dedicated server capacity

This section discusses the processing capabilities for the V5R1 Dedicated Servers. Table 12-2 shows the performance capacities for the five V5R1 iSeries Dedicated Server processor features.

Table 12-2 Performance capabilities of V5R1 Dedicated Server for Domino servers

Processor feature	Mail & Calendaring Users (MCU)	Processor CPW “non-Domino”	Processor CIW	Interactive CPW
270-2452	3,070	100	380	0*
270-2545	5,050	240	825	0*
820-2456	3,110	120	385	0*
820-2457	6,660	240	825	0*
820-2458	11,810	380	1,590	0*

The following sections describe the performance capacities shown in Table 12-2.

Mail and Calendaring Users (MCU)

Mail and Calendaring Users is a workload metric for comparing Domino capacity across different Domino server platforms. Each Mail and Calendaring user completes the following actions an average of every 15 minutes except where noted:

As a rule, when equating mail & calendaring users to real world users, divide by two.

- ▶ Opens mail database, which contains documents that are 10K bytes in size.
- ▶ Opens the current view.
- ▶ Opens five documents in the mail file.
- ▶ Categorizes two of the documents.
- ▶ Composes two new mail memos/replies 10K bytes in size (every 90 minutes).
- ▶ Marks several documents for deletion.
- ▶ Deletes documents marked for deletion.
- ▶ Creates one appointment (every 90 minutes).
- ▶ Schedules one meeting invitation (every 90 minutes).
- ▶ Closes the view.

Processor Commercial Processing Workload (CPW)

Commercial Processing Workload describes the amount of non-Domino processing that is available on the Dedicated Server for Domino for applications that are not Domino-based and are not used in conjunction with Domino-based applications. The processor CPW represents a planning estimate, not a guaranteed level of capacity to perform non-Domino work. As discussed earlier, non-Domino processing is considered complementary when it is used in conjunction with Domino-based processing. Throughout the remainder of this chapter, we use the terms non-Domino and non-Domino processing to refer to processing that is subject to the Processor CPW rating.

You can think of the non-Domino capacity (Processor CPW) of a Dedicated Server in the following ways:

- ▶ Non-Domino work should not exceed 15% of the total capacity of the processor. This is the amount the Processor CPW guidelines state for all Dedicated Server for Domino models.

- ▶ Non-Domino work on the Dedicated Server may consume up to the full capacity of Processor CPW. When the maximum Processor CPW capacity is exceeded, the response times of non-Domino processes increase since they contend for a fixed and saturated Processor CPW capacity. System overhead is generally not observed except for brief periods of time, typically when fluctuating non-Domino work is occurring.
- ▶ Executing non-Domino work in conjunction with Domino-based processing allows the system to treat the non-Domino work as complementary work. Since complementary work is not subject to the 15% non-Domino guideline, the work can use the full processor capability of the system.

Processor Compute Intensive Workload (CIW)

Compute Intensive Workload values are application compute-intensive projections based on the characteristics of workloads like SAP, Domino Mail and Calendaring Users and Domino applications, and typically Java WebSphere.

The Processor CIW rating was introduced in V5R1. It provides a better metric than CPW for comparing iSeries servers with respect to compute intensive workloads such as Domino processing. The *V5R1 Performance Capabilities Reference* at <http://publib.boulder.ibm.com/pubs/html/as400/online/chgfrm.htm> contains CIW ratings for all V5R1 iSeries models as well as additional information about the metric itself. The CIW is meant to depict a workload that has the following characteristics:

- ▶ The majority of the system processing time is spent in the user application instead of system services. For example, typical Domino applications spend about 80% of the total processing time in application code (Domino code).
- ▶ Compute-intensive applications tend to be considerably less I/O intensive than the commercial application processing depicted by CPW. Therefore, cache miss rates are low and I/O contention is minimal.

Interactive CPW

Interactive CPW (often referred to as green screen processing) is designed to support system administration functions using a 5250 session. The Dedicated Server for Domino Models 270 and 820 are defined as having an Interactive CPW of 0.

In practice, a single interactive job running on these models will have access to up to 15% processing capacity of the Processor CPW while performing system administration activities. However, if two or more interactive jobs are simultaneously using CPU resource, then all interactive jobs will contend for the Interactive CPW capacity that is effectively 0 and will experience increased response times. If a single interactive job is doing significant input/output to the display device (which is not characteristic of system administration activities), that interactive job will conform to the Interactive CPW capacity, which is effectively 0.

Tip: We recommend that you use Operations Navigator to perform as much as possible of the administration work. Since the workload caused in the iSeries server by Operations Navigator is not interactive work, it has the full processor capacity available as long as a Domino server is active and up to 15% in situations where all Domino servers have been ended.

Note: Mail & Calendar Users, Processor CPW, Processor CIW, and Interactive CPW capacities are not additive. Each of the guidelines in Table 12-2 on page 348 individually represent a recommended maximum capacity.

12.3 Evaluating iSeries application integration on Dedicated Server for Domino

Many of the questions we have received about appropriate workloads for the Dedicated Server focused on application and database integration. The V5R1 Dedicated Server for Domino enhancements have made this issue much less complex by providing significantly improved support for application integration. This section provides general guidance and analysis to help you determine whether your proposed workload mix can be expected to take full advantage of the Dedicated Server for Domino capacity and whether it falls within the 15% DB2 database processing guideline.

Figure 12-1 shows the common techniques for integrating Domino applications with DB2 UDB database files. It classifies the methods as to whether they are generally considered Domino-based, DB2 database, or Java processing on the Dedicated Server. From a programming standpoint, these integration techniques are the same on the iSeries as they are on other Domino platforms.

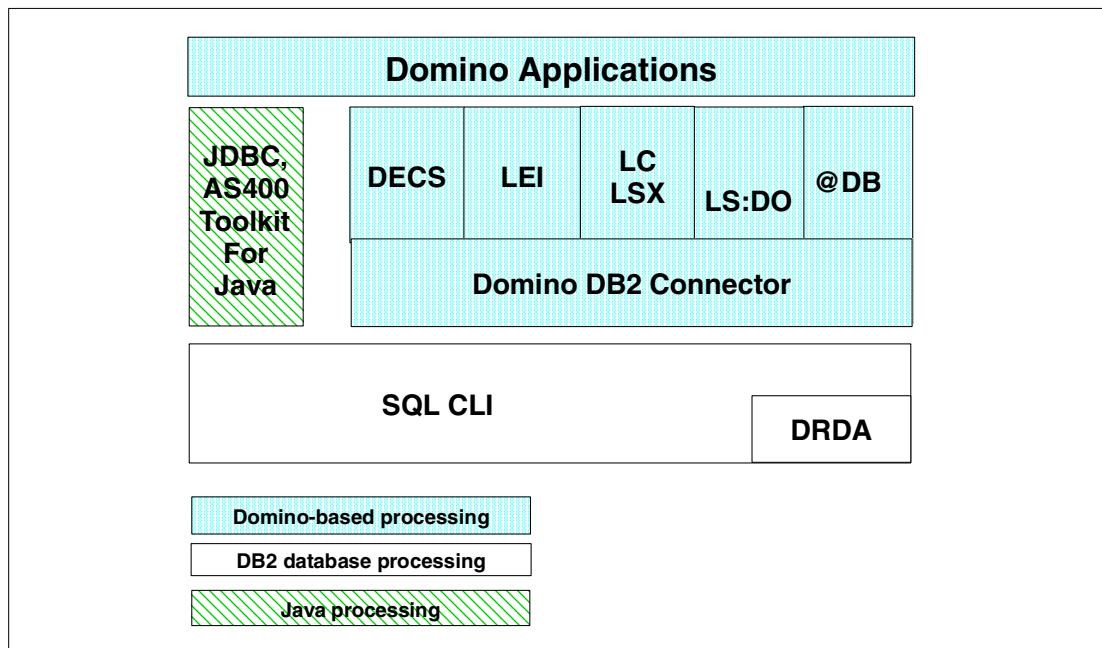


Figure 12-1 Domino enterprise integration techniques

Under the covers, most of the methods shown use the OS/400 SQL Call Level Interface (CLI) to access DB2 data. The AS/400 Toolkit for Java uses the Licensed Internal Program Interface (LIPI) by default. But for local DB2 database access, it can be directed to use the CLI interface. To access remote data, the Toolkit uses the LIPI interface along with QZDA processing rather than the CLI and DRDA path. (Another possible exception would be some custom-written connectors.)

When the data being accessed resides on a different server in the network, OS/400 uses the DRDA layer to provide the access. Again, this happens under the covers without extra coding by the Domino programmer.

The sections that follow discuss each of these methods and their likely performance implications on the Dedicated Server in more detail. But first, here is more detail on what is considered DB2 database processing.

12.3.1 DB2 database processing defined

Since the Dedicated Server has a 15% guideline for all DB2 database processing, it is important to understand what exactly is considered DB2 database processing by the system and to consider some methods for optimizing DB2 database processing.

A V4R5 enhancement to OS/400 provided reporting capability that shows the percent of CPU time spent in DB2 database processing. This mechanism can distinguish between processing time spent in application code versus when the application accesses database records in DB2 database files. Only the processing time in database code and access methods is considered DB2 database processing. Processing time in SQL, Query/400, DRDA, and other database access methods is generally considered DB2 database processing.

The small amount of processing time to establish connections and do database-related work, such as obtaining time stamps, is not considered DB2 database processing. Generally, the vast majority of processing in these methods is related to accessing database records. Processing time in JDBC and ODBC is described in Figure 12-1 as Java processing because it technically does not count as DB2 database processing for a workload until control is passed to an SQL-type layer.

To determine how much system resource is being consumed by DB2 database processing, refer to 12.4, "Observing Dedicated Server performance" on page 362, 4.10.4, "Work with System Activity (WRKSYSACT) command" on page 71, and Figure 4-8 on page 72.

For information on optimizing your application's use of DB2 database processing, refer to the iSeries Information Center for V5R1 at:

<http://publib.boulder.ibm.com/html/as400/v5r1/ic2924/index.htm>

At this site, click the following links:

1. **Database and file systems** in the left column
2. **DB2 UDB for iSeries**
3. **Manuals and redbooks**
4. **Database Performance and Query Optimization**

Domino Enterprise Connection Services (DECS)

Domino Enterprise Connections Services runs as part of the Domino server. DECS allows access to backend databases in real time mode for Domino users. The data that a Domino user sees looks like it is coming from a Domino database. In fact, it is coming from a relational database outside Domino. This access happens through a connector, which in our case is most likely DB2, but may also be a custom-written connector or a premium ERP connector. The amount of DB2 database processing required varies based on the design of the connector being used by DECS.

We examine some DECS scenarios to show how the DB2 processing varies depending on the environment. All of the environments have the Domino server and the DB2 relational database on a Dedicated Server for Domino system. The database access is done locally; no DRDA connectivity is involved.

DB2 DECS scenario 1 is a good fit for the Dedicated Server because of its minimal use of DB2 database processing. The second and third examples, DB2 DECS scenario 2 and Custom written DECS Connector scenario, might require more DB2 database processing than what is available on the Dedicated Server for Domino depending on how heavy the DB2 database processing requirements are. Hopefully these scenarios will help provide some guidance when deciding if a Dedicated Server for Domino is appropriate for inclusion of DECS for DB2 database access.

DB2 DECS scenario 1

In our first example, we have a Domino application that is exposing DB2 UDB records to application users via a DB2 DECS connection. Only one relational table is involved and the connection is through the DB2 connector. The amount of processing involved for DB2 access is quite minimal, so this scenario is a good fit for the Dedicated Server for Domino system.

The application overall performs relatively simple DB2 processing that does not require significant processor resources. The various steps involved in a transaction originating from the application user are outlined in the following list. For each step, we highlighted which processing steps are Domino-based processing versus DB2 database processing:

1. (Domino-based) Domino user opens a document.
2. (Domino-based) Domino server receives a request and passes it to DECS.
3. (Domino-based) DECS sends a request to DB2 connector to find fields in the DB2 table.
4. (DB2 database) DB2 receives a request.
5. (DB2 database) DB2 finds a row.
6. (DB2 database) DB2 returns fields to DECS.
7. (Domino-based) DECS receives fields from DB2.
8. (Domino-based) DECS sends fields to the Domino document that the user is viewing.

Because DB2 UDB is integrated on iSeries and because this scenario involves simply fetching or possibly inserting a single row, this use of DECS is not likely to exceed the DB2 database processing limits on the Dedicated Server.

DB2 DECS scenario 2

The second scenario we examine is quite similar to the first one. However, there is more use of DB2. Again we are exposing DB2 UDB records to a Domino application user via DB2 DECS connection. This time we access information from multiple DB2 relational tables. The increase in the number of tables connecting into increases from 1 in the first scenario to 10 in this scenario.

This application is more complicated in that a single document is populated with multiple DB2 tables. Each DB2 table requires a separate connection from DECS to be active to work with the data in that table. This scenario requires much more DB2 database processing because of the multiple table access and therefore is not appropriate for the Dedicated Server for Domino server. More than 15% of the CPU will be utilized for DB2 access, not allowing the application to fully use the capability of the Dedicated Server for Domino.

The various steps involved in a transaction originating from the application user for this scenario are outlined here. As before, each step indicates which is Domino-based processing and which is DB2 database processing:

1. (Domino-based) Domino user opens a document with fields that will be populated from multiple DB2 tables.
2. (Domino-based) Domino server receives a request and passes it to DECS.
3. (Domino-based) DECS sends multiple requests to a DB2 connector to find fields in DB2 tables.
4. (DB2 database) DB2 receives requests.
5. (DB2 database) DB2 finds a row selected in each table.
6. (DB2 database) DB2 returns fields from each table to DECS.
7. (Domino-based) DECS receives fields from DB2.
8. (Domino-based) DECS sends fields to the Domino document the user is viewing.

This scenario is not appropriate for the Dedicated Server for Domino server because too much time is spent doing DB2 database access. The number of tables involved to populate the document are too numerous to stay within the 15% CPU guideline.

Custom written DECS Connector scenario

In this scenario, we use a custom-written connector written by business partner AppXYZ. The AppXYZ connector interfaces Domino with its custom database environment. Instead of using DECS access data directly in DB2 UDB, DECS will interface with the business rules that are required when working with the AppXYZ application data. This data is in the format of people and accounts rather than direct DB2 tables. The data that represents these people and accounts is generated from DB2 tables, but there are business rules that need to be followed when working with the various accounts and people in the AppXYZ application. In this scenario, to return values of fields in the database, DECS sends requests to the AppXYZ connector to generate a report.

The steps that are involved in a DECS connection using the AppXYZ connector are outlined here. Each step specifies Domino-based processing versus DB2 database processing:

1. (Domino-based) Domino user opens a document.
2. (Domino-based) Domino server receives a request and passes it to DECS.
3. (Domino-based) DECS sends the request to AppXYZ connector to find a field in the AppXYZ application. That is, the Domino application is requesting information about an AppXYZ account. The AppXYZ connector will ensure the proper business rules are followed when accessing the underlying DB2 tables that comprise the account information being requested.
4. (Complementary) AppXYZ connector receives a request for account information.
5. (Heavy DB2 database) AppXYZ connector uses the field request to collect records from DB2 that represent the account. In addition, the connector creates a log entry of the query results and sends a report to a printer.
6. (Complementary) AppXYZ connector returns the account information back to DECS.
7. (Domino-based) DECS receives fields from the AppXYZ connector.
8. (Domino-based) DECS sends fields to the Domino document that the user is viewing.

The AppXYZ connector described above may not perform well on the Dedicated Server. The complexity of the work this connector performs in DB2 database processing may require more than the 15% CPU guideline. Therefore, it is not recommended for implementation on a Dedicated Server for Domino server.

Information about determining the amount of DB2 database processing on a system is provided in 12.4, "Observing Dedicated Server performance" on page 362.

To ensure that an application consistently takes full advantage of the Domino price/performance capabilities of the Dedicated Server, we recommend that you use Domino database integration between the Dedicated Server and DB2 databases on another iSeries or AS/400 server. This implementation involves DRDA connectivity, which requires very little DB2 database processing on the requesting Dedicated Server for Domino server.

Lotus Enterprise Integrator (LEI)

Lotus Enterprise Integrator, known previously as NotesPump, is a method for replicating or transferring the contents of one database to another. The source and target database can be any combination of supported formats, including Domino, DB2, SAP, etc. Like DECS, LEI uses a connector to interface the databases. Here are the recommendations for using LEI on the Dedicated Servers:

- ▶ Generally, when at least one of the databases in the transfer is a Domino database, DB2 database processing will be within the 15% CPU guideline. Unless you have multiple transfers occurring simultaneously, this may push you above the 15% CPU guideline. In this case, the transfers will still complete but may take a bit longer since the DB2 database processing will be limited to 15% of the CPU.
- ▶ When both the source and target databases are something other than Domino databases, it is likely that the DB2 database processing *will exceed* the 15% guideline. Consequently, the performance you experience for the data transfer will be less than the full performance capabilities of your Dedicated Server.
- ▶ Many customers want to perform large-scale LEI data transfers infrequently, during off-shift hours. In these cases, the 15% CPU guideline *will be exceeded*. However, due to the off-shift processing, the reduced performance that results from exceeding DB2 database performance limits during the data transfer is acceptable. The data transfers will run successfully; they will simply not be able to take full advantage of the performance capabilities of the Dedicated Server.

Note: The Real-time component of LEI has the same characteristics as DECS.

12.3.2 Accessing external databases on the same iSeries server

An external database is a database that is not a Domino.NSF file, such as a DB2 database file. From Domino, you can access an external database by using @DB Functions or agents written in LotusScript or Java (via JDBC).

LotusScript access

LotusScript has been the long running programming language of choice for the majority of Domino application. As such, there are rich API sets available to choose from when coding your Domino application in LotusScript to access external data sources. The LotusScript programming language in Domino provides two sets of classes for accessing backend resources – LS:DO and LC LSX.

LotusScript:Data Object (LS:DO) is a set of classes that allow easy programming access to external databases. LS:DO offers an ODBC-like programming interface. The API set is composed of three classes that make access to backend resources very easy to program.

Lotus Connector Lotus Software Extension (LC LSX) is a set of classes that allow for a consistent programming interface to multiple backend systems and disparate data sources. The LC LSX set of APIs provides the ultimate flexibility in programming your Domino application to access many different types of data sources on different backend systems. Using the LC LSX classes, it is very easy to program an application that accesses DB2 data on the iSeries server and that accesses Oracle data on a Sun system.

On the iSeries, the underlying access from LS:DO or LC LSX occurs in two ways, both of which have components that are considered DB2 database processing:

- ▶ When the Domino LS:DO or LC LSX processing is requested from a browser interface, OS/400 provides direct access to DB2 UDB using the SQL Call Level Interface (CLI).
- ▶ For Notes client applications, LS:DO or LC LSX access is through an ODBC connection to access external database files.

Processing time in SQL CLI and ODBC will be considered DB2 database processing.

Java access

Java is the other major programming language to choose from when writing your Domino applications. If you choose Java, you are opening your application up to applets, servlets, and stand-alone Java applications, in addition to agents and script that execute directly in your application. The Java interface to backend relational databases is *Java Database Connectivity (JDBC)*. JDBC is a defined interface that allows Java programs to access external data, such as DB2 UDB databases.

There are two different JDBC drivers that can be used on the iSeries platform:

- ▶ Native JDBC driver
- ▶ iSeries Toolkit for Java JDBC driver

The JDBC driver classes are considered DB2 database processing and thereby will need to follow adhere to the 15% CPU guideline to enjoy the full performance capabilities of the Dedicated Server for Domino server.

Java programs, in general, when used in conjunction with Domino-based applications, are considered complementary processing. So, the non-JDBC processing that is done in your application, servlet, or agent, will be considered complementary. Only the JDBC classes execution portion of the code will be considered DB2 databases processing.

So you may ask, “How much external database access is OK?” We have found that simple queries from a Domino application using JDBC access to DB2 databases on the same Dedicated Server should perform acceptably. Factors that could potentially push this DB2 database processing above the 15% limit include:

- ▶ Running complex queries that are processor intensive
- ▶ Running many queries simultaneously
- ▶ Running queries against extremely large DB2 databases
- ▶ More complex database access

As mentioned in the DECS discussion, to ensure that an application consistently takes full advantage of the Domino price/performance capabilities of the Dedicated Server, we recommend using the Domino database integration between the Dedicated Server and DB2 databases on another iSeries or AS/400 server, or keeping DB2 database access light.

12.3.3 Accessing external databases on a different iSeries server

When a Domino application accesses an external database on another server, you use OS/400 *Distributed Relational Database Architecture (DRDA)* support. From a programming perspective, you use one or more of the normal Domino techniques for accessing relational data: @db commands, LS:DO, LC LSX, JDBC, 00 DECS, or LEI. Under the covers, these techniques use OS/400 DRDA support when the relational data resides on another server.

The DRDA call, which occurs on your Dedicated server, is DB2 database work. However, all of the DB2 work to access the database and return the result runs on the connected server that houses the database. Therefore, the processor load on the Dedicated Server to send DRDA database requests to another machine is light and should generally perform well on the Dedicated Server, requiring less than the 15% DB2 database guideline.

As mentioned in the explanation that followed Figure 12-1 on page 350, the AS/400 Toolkit for Java uses the LIPI interface by default. It uses QZDA processing to access DB2 database resources on a remote system. This processing is also considered DB2 processing.

Although standard DRDA access will perform acceptably on the Dedicated Server, extensive use of the following factors has the potential to increase the SQL CLI component and push your DB2 database processing above the recommended 15% guideline:

- ▶ **Large data streams being returned:** This causes increased buffer management work.
- ▶ **Character translation:** Occurs on the source server (the Dedicated Server). Character translation occurs when manipulating string data types. iSeries is an EBCDIC-based system and Domino is LMBCS-based. String data needs to be converted when working with native string data and Domino string data. This character conversion is performed automatically by many of the integration options.
- ▶ **Two-phase commitment control:** Is coordinated by the source server (the Dedicated Server).
- ▶ **Large number of transactions per request:** More SQL DRDA transactions require more processing on the source. If the application is doing many queries or inserts, each query or insert requires an invocation of the SQL CLI and DRDA code.
- ▶ **Buffer format:** If the remote system is not an iSeries or AS/400 (or if it is an iSeries or AS/400 but the data tends to have large amounts of NULLABLE or variable length character (VARCHAR) fields), then the formatting type used is less efficient and requires more processing on the source.

12.3.4 Java and Domino integration

Java can be used in many ways on an iSeries server. With the V5R1 Dedicated Server for Domino enhancements, Java processing is considered complementary processing when run in conjunction with a Domino-based application. If the Java processing is being invoked from a Domino application, then the processing time in Java will be complementary processing. If the Java processing is invoked from a source other than a Domino application, then as long as the processing is used in conjunction with Domino processing, it will be considered complementary. Be sure you understand that when Domino processing is not present, Java workloads will be allowed to use only 15% of the CPU resource.

As noted in the discussion about JDBC data access, with the new V5R1 Dedicated Server for Domino enhancement, Java processing will have access to the full capability of the Dedicated Server for Domino resources, assuming it does not use more than 15% CPU for DB2 database processing. External database access using JDBC is DB2 database processing. Keeping the database access light and hosting the DB2 database on a non-Dedicated Server for Domino system will increase the chances your Java code will perform well on the Dedicated Server.

When sizing a new Java workload, IBM Workload Estimator for iSeries (see 12.6, “Sizing a Dedicated Server for your Domino application” on page 368) can be used to project the amount of DB2 database processing that will be required. The Workload Estimator can project DB2 database utilization for workloads, such as Java and WebSphere, in addition to Domino. For an existing Java workload, see “Observing DB2 database processing” on page 363 to determine how much DB2 database resource is currently being consumed.

12.3.5 WebSphere and Domino integration

Like Java processing, with the V5R1 Dedicated Server for Domino enhancements, WebSphere processing such as WebSphere Application Server and WebSphere Commerce Suite is now considered complementary processing when used in conjunction with Domino-based processing. Similar to what we described for Java in the section above, be sure you understand that when Domino processing is not present, WebSphere processing

will be allowed to only use 15% of the CPU resource. If you plan to run WebSphere applications independently of Domino, then you should consider an iSeries server with standard or base processors to ensure that you have constant access to the full capability of the server when Domino processing is not present.

There are two basic techniques for integrating Domino and WebSphere. In the first case, you would use Domino's HTTP support as your primary HTTP server and connect to WebSphere Application Server as needed for supporting function such as JavaServer Pages (JSP) and Enterprise JavaBeans (EJB).

In the second case, you would use IBM's HTTP support as the primary HTTP server to support WebSphere processing and only connect to Domino to handle requests involving Domino objects. In both cases, WebSphere processing and Domino processing run under separate job structures, and WebSphere processing will be considered complementary when run in conjunction with Domino.

When using WebSphere Commerce Suite as part of a WebSphere solution on a Dedicated Server, it is essential that your DB2-based catalogue reside on a server other than a Dedicated Server for Domino. If the WebSphere Commerce Suite catalogue in DB2 is accessed locally, it is likely that you will exceed the 15% DB2 database processing guideline either right away or as your application grows. If you are designing a new application that includes Domino and WebSphere integration, you should use the IBM Workload Estimator for iSeries (see 12.6, "Sizing a Dedicated Server for your Domino application" on page 368). The Workload Estimator can help you determine whether the Dedicated Server for Domino has adequate DB2 database processing capacity to handle the projected amount of DB2 database activity.

12.3.6 File serving on a Dedicated Server

Many customers have inquired as to whether it is feasible to perform file serving processing on a Dedicated Server in addition to their Domino applications. With the Dedicated Server for Domino enhancements in V5R1, file serving processing is considered complementary and has access to the full CPU capacity of the Dedicated Server for Domino when the file serving activity is performed when Domino processing is active. When Domino processing is not active, file serving processing is subject to the 15% non-Domino CPU limit.

We have found in this scenario that generally more than enough CPU processing is available to handle your file serving requirements, even though Domino is not running in conjunction with file serving. Please consider standard file serving guidelines to help ensure you have sufficient capacity on your Dedicated Server for Domino to perform your required file serving processing. When sizing a Dedicated Server to include file serving, you should take into account that you might need more disk arms to accommodate the I/O activity generated by the file server. File serving performed by an Integrated xSeries Server or with iSeries NetServer requires very little DB2 database processing and is not a concern for exceeding the 15% DB2 database guideline.

Sizing for Integrated xSeries Server

Each Integrated xSeries Server requires some iSeries processor resources. Therefore, you should not plan on supporting a large number of Integrated xSeries Servers on a Dedicated Server. Chapter 17 of *iSeries Performance Capabilities Reference Version 5 Release 1* provides information about the performance characteristics of the Integrated xSeries Server, as well as for the Integrated Netfinity Server. It also includes information on the new 850 MHz PCI version of the Integrated xSeries Server. You can find this document online in the iSeries library at: <http://publib.boulder.ibm.com/pubs/html/as400/online/chgfrm.htm>

As a rule of thumb, one to two Integrated xSeries Servers can be supported on Dedicated Server for Domino with a 1-way processor, two to three on a Dedicated Server for Domino with a 2-way processor, and possibly up to four on the Dedicated Server for Domino Model 820 with a 4-way processor.

Chapter 17 in *Performance Capabilities Reference V4R5* indicates that you should plan approximately 32 CPW for each 700 MHz Integrated xSeries Server that is doing fairly heavy file serving. This CPW requirement is due to the I/O that is being generated from the xSeries server. These I/O requests consume a certain amount of CPW to process. The CPW requirement can be as high as 45 in a worst case heavy file serving scenario. An Integrated xSeries Server that is running applications may require up to 50 CPW. With the 850 MHz Integrated xSeries Server, the CPW requirements may be even higher since this faster file server can generate a large number of I/O requests to the Dedicated Server for Domino server.

Sizing for iSeries NetServer

If you have an existing environment running NetServer (OS/400 Support for Windows Network Neighborhood), the most accurate way to size additional NetServer traffic, or to size it on another server, is to determine how many CPW are required for NetServer processing.

Use the performance monitor or another technique to determine how much CPU resource is being used over a representative time period, for example 30 minutes, by the following processes:

- ▶ SMBWORKER
- ▶ SMBREQMONITORPKT
- ▶ SMBCONNWORKER
- ▶ SMBCONNMONITOR
- ▶ SMBSERVERMAIN
- ▶ QZLSSERVER
- ▶ QZLSFILE

Note: IPTRTRxxx tasks are also used for NetServer processing, but it may not be possible to separate out the portion due to NetServer versus other TCP/IP traffic.

Let's look at an example. If the SMB and QZLS processes use 5% CPU of a Model 720-2062, which has a Processor Rating of 420 CPW, then you can project that the workload measured requires approximately 21 CPW for NetServer processing ($420 * 0.05 = 21$). Use this number to compare with the Processor CPW (non-Domino) rating for a Dedicated Server for Domino. As long as the Processor CPW on the Dedicated Server for Domino is greater than 21, the Dedicated Server for Domino can accommodate the NetServer workload.

Based on informal tests in the laboratory, it was found that when using a simulated file serving workload to test NetServer, approximately 7000 bytes per second could be processed per CPW. To scale this up for multiple clients, 50 CPW would handle approximately 350,000 bytes per second ($7000 * 50 = 350,000$) of NetServer file serving traffic. This may be another way to estimate your needed CPW requirements for NetServer if you have an idea of the expected workload in terms of document sizes and data transfer rates.

12.3.7 Backup Recovery and Media Services on a Dedicated Server

Backup Recovery and Media Services (BRMS) on iSeries provides the following capabilities:

- ▶ Media management
- ▶ Automation of backup
- ▶ Simplification of recoveries

- Hierarchical storage management
- Tape library support

It includes the ability to save databases from a Domino server (or a QuickPlace or SameTime server) while the databases are in use. For additional information on using BRMS with Domino, see 6.8, “Online backup with BRMS” on page 178, 13.4.1, “Incremental online backup with BRMS” on page 384, and the Online Lotus Server Backup site at:

<http://www.ibm.com/eserver/iseries/service/brms/domino.htm>

This section provides information and results based on informal testing of BRMS backup function on a Dedicated Server for Domino Model 270-2423 with OS/400 V4R5. BRMS backup has a range of performance behaviors that depend on the options selected for backup. We tested three scenarios using the STRBKUBRM CTLGRP(QLTSSVR) SBMJOB(*NO) command. All tests had the following characteristics:

- Tests were performed with a 6386 ¼" tape drive (13 GB) with a specified density of *QIC5010.
- The objects being saved were primarily 9 MB mail databases.
- Each test was approximately one hour in duration.
- Each test issued the STRBKUBRM command from the system console (an interactive session).

In addition, the notes.ini SAVDOMBRM_FILES_IN_GROUP variable was utilized to specify the number of objects that were saved in a group. The SAVDOMBRM_FILES_IN_GROUP variable controls how many objects are bundled together for a save operation. The larger the number is, the larger the blocks of objects are that are written out to tape. This variable can be used to optimize save times by blocking the number of separate writes to tape.

- Scenario 1: Backup all Domino servers saving *five* objects at a time, servers *not active*
- Scenario 2: Backup all Domino servers saving *five* objects at a time, servers *active*
- Scenario 3: Backup all Domino servers saving *120* objects at a time, servers *not active*

The test results are reflected in Table 12-3.

Table 12-3 BRMS backup performance results

Scenario	CPU utilization	SAVDOMBRM_FILES_IN_GROUP notes.ini variable
1	17.3%	5
2	19.9%	5
3	3.5%	120

Based on the test results, the following observations can be made:

- Performing backup with fewer objects at a time resulted in shorter waits for the tape, more frequent use of the CPU lasting a few seconds, and a higher average CPU utilization during the backup.
- Backing up the servers while active used approximately 15% more CPU as shown based on the results of scenarios 1 and 2.
- Specifying SAVDOMBRM_FILES_IN_GROUP=120 in the notes.ini files of the servers for scenario 3 provided a more efficient use of CPU resource to perform the backup. Backing up a large number of files at a time, such as 120, resulted in a longer wait for the tape resource between the times when CPU was used.

When the CPU was used, it tended to be for many seconds at a high utilization followed by long wait for the tape with a low overall CPU utilization.

Note: While the overall CPU utilization was lower for saving 120 files in a group, there were shorter CPU spikes that did occur during the save operation that could have impacted end user response time of other jobs running on the server.

- Using a faster tape drive such as a 3590 should decrease the amount of tape wait time. A decrease in tape wait time will, therefore, result in an increase in the amount of CPU that will be requested by the STRBKUBRM command.

Online backup recommendations

When BRMS backup is the only interactive job on the system, it is not limited by the zero-interactive rule. If other interactive jobs are active simultaneously with the backup operation, all interactive jobs including BRMS backup will experience increased response times and will approach the Interactive CPW capacity, which is effectively 0.

This should not impact Domino-based processing or complimentary processing. Performing backup during periods with the fewest number of users active reduces the performance impact to attached users. It also allows the backup operation to run as efficiently as possible. BRMS processing in this case is complementary because Domino processing is present.

Offline backup recommendations

If all Domino processing is ended on the Dedicated Server for Domino, the offline backup will be subject to the non-Domino 15% CPU processing guideline. In the case when one Domino server is being backed up offline while another Domino server on the same system is still active, the offline backup processing is considered complementary processing.

Note that while the non-Domino processing guideline is 15% CPU utilization, we observed a slightly higher utilization during our tests. The backup operation might experience increased response times if the backup operation exceeds the Processor CPW guidelines. You can control the amount of CPU used by BRMS backup by using OS/400 job priority to set the priority of the BRMS backup lower than other jobs. This action can lessen the impact to Domino processing on the system.

12.3.8 Logical partitioning (LPAR) on a Dedicated Server

With V5R1, iSeries logical partitioning is supported on both the Model 270 and Model 820 non-Dedicated Server for Domino and Dedicated Server for Domino servers. iSeries logical partitioning lets you run multiple independent servers, each with its own processors, memory, disk and operating system, within a single symmetric multiprocessing iSeries. It also provides special capabilities such as having multiple versions of OS/400, multiple versions of Domino, different system names, languages, and time zone settings. For additional information on logical partitioning on the iSeries, see the logical partitioning site at: <http://www.ibm.com/eserver/iseries/lpar>

iSeries LPAR is different from running multiple Domino partitions (servers) on the same physical system. It is *not* necessary to use iSeries LPAR to run multiple Domino servers on an iSeries server.

When you use LPAR with a Dedicated Server, the Dedicated Server for Domino CPU processing guidelines are pro-rated for each logical partition based on how you divide the CPU capability. For example, suppose you use iSeries logical partitioning to create two logical partitions, and specify that each logical partition should receive 50% of the CPU resource.

From a Dedicated Server for Domino perspective, each logical partition runs independently from the other, so you will need to have Domino-based processing active in each logical partition in order for non-Domino work to be treated as complementary processing. Other Dedicated Server for Domino processing requirements, such as the 15% DB2 processing guidelines and the 15% non-Domino processing guideline, will be divided between the logical partitions based on how the CPU was allocated to the logical partitions.

In our example above with 50% of the CPU resource in each logical partition, the DB2 database guideline will be 7.5% CPU utilization for each logical partition. The non-Domino processing guideline would be divided similarly. Only Domino and complementary Domino work runs independently in each logical partition. All other work is totalled across all logical partitions on the single symmetric multiprocessor iSeries.

12.3.9 Linux on a Dedicated Server for Domino

As with other iSeries servers, to run Linux on a Dedicated Server for Domino, it is necessary to use logical partitioning. Because Linux is its own unique operating environment and is not part of OS/400, Linux needs to have its own logical partition of system resources, separate from OS/400. The iSeries Hypervisor allows each partition to operate independently. When using logical partitioning on iSeries, the first logical partition, the primary partition, must be configured to run OS/400. For more information on running Linux on iSeries, see:

<http://www.ibm.com/eserver/iseries/linux>

You may also reference Chapter 13, "iSeries Linux Performance" in the *AS/400 Performance Capabilities Reference Version 5 Release 1* document. You can find this document online in the iSeries library at:

<http://www.publib.boulder.ibm.com/pubs/html/as400/online/chgfrm.htm>

Running Linux in a Dedicated Server for Domino logical partition will exhibit different performance characteristics than running OS/400 in a Dedicated Server for Domino logical partition. As described in 12.3.8, "Logical partitioning (LPAR) on a Dedicated Server" on page 360, when running OS/400 in a Dedicated Server for Domino logical partition, the Dedicated Server for Domino capacities, such as the 15% DB2 processing guideline and the 15% non-Domino processing guidelines, are divided proportionately between the logical partitions based on how the processor resources were allocated to the logical partitions.

However, for Linux logical partitions, the Dedicated Server for Domino guidelines are relaxed, and the Linux logical partition can use all of the resources allocated to it, outside the normal guidelines for Dedicated Server for Domino processing. This means it is not necessary to have Domino processing present in the Linux logical partition. In addition, all resources allocated to the Linux logical partition can essentially be used as though it were complementary processing. It is not necessary to proportionally increase the amount of Domino processing in the OS/400 logical partition to account for the fact that Domino processing is not present in the Linux logical partition.

By providing support for running Linux logical partitions on the Dedicated Server, it allows customers to run Linux-based applications, such as Internet firewalls, to further enhance their Domino processing environment on iSeries. At the time this publication was written, a version of Domino did not exist that is supported for Linux logical partitions on iSeries.

12.4 Observing Dedicated Server performance

With the introduction of complementary behavior in V5R1, the task of managing Domino versus non-Domino processing to a 3 to 1 ratio is no longer required. When Domino-based processing is present on the system, other processing is treated as complementary and has access to the full capacity of the Dedicated Server. However, observing how system resources are being used by applications can assist with system administration and help you plan for future growth. This section discusses the use of the Management Central function of Operations Navigator and other tools to observe Dedicated Server for Domino performance.

The Management Central function of Operations Navigator uses a client interface to provide real-time observation of performance metrics such as CPU utilization, disk utilization, and paging and faulting rates. Beginning with V4R5, a new metric called *Secondary CPU Utilization* is available that is specific to the Dedicated Server. It shows the amount of non-Domino processing that is occurring on the system. We explain this metric in more detail below. To provide a good indication of overall system performance, you can select and then view multiple metrics simultaneously in real time through Management Central. For additional information on getting started with Operations Navigator and Management Central functions, see the iSeries Information Center at:

<http://publib.boulder.ibm.com/pubs/html/as400/v5r1/ic2924/index.htm>

Once you reach this site, click **Operations Navigator** in the left column and then **Management Central**.

You can also find more details on how to use Management Central to collect performance statistics for your Domino environment in 4.12, "Management Central performance monitor" on page 89.

12.4.1 Operations Navigator: Management Central

Operations Navigator provides a graphical view into how your system is performing. System monitors gather and present real-time performance data. You can choose from a wide variety of system metrics to be displayed simultaneously. We use Management Central to display information about our Domino environment to see how much complementary processing and non-Domino work is happening.

Observing non-Domino and complementary processing

Following is the official description of the Management Central *Secondary Workload* utilization metric:

CPU Utilization (Secondary Workloads): *This metric can be used on the Dedicated Server for Domino to see how much non-Domino work is being done on the system. Because the Dedicated Server for Domino is intended to be used as a server devoted to Domino work, this metric helps you to identify and manage those workloads not directly contributing to that primary system activity. As with interactive utilization, managing secondary workloads helps you to maintain optimal system performance.*

The Secondary Workload metric shows processing on the Dedicated Server for Domino that is not Domino-based. When non-Domino processing is active without Domino-based processing, non-Domino processing can use up to the allowed guideline which is 15% CPU utilization. In practice, you might observe slightly higher utilizations for this kind of processing. The Secondary Workload metric will show the actual amount of CPU processing that is non-Domino in nature.

The Dedicated Server for Domino is designed so that Domino-based processing should be present on the system. In this case, non-Domino processing is considered complementary and is allowed use of the full capability of the Dedicated Server for Domino. When Domino-based processing is present, the Secondary Workload metric will show complementary processing that may be significantly more than 15% since it is allowed access to the full capability of the system.

Here is an example. Suppose we have a Dedicated Server for Domino environment with Domino-based processing present, and the complementary processing shows 40% CPU as reported by the Secondary Workload metric. If the Domino-based processing is removed, the non-Domino processing that was being treated as complementary now becomes non-Domino processing (which is allowed up to 15% CPU if Domino-based processing is not present) and the Secondary Workload metric would show this processing dropping from 40% CPU to something closer to 15% CPU. It is because of this potential for an application's CPU resources to be reduced that we recommend that only applications in support of Domino or applications that require no more than 15% of the CPU be deployed on the Dedicated Server for Domino in addition to Domino-based applications.

Observing DB2 database processing

Another metric called *Database Capability* is available for observation using Management Central. The Database Capability metric has the following definition:

CPU Utilization (Database Capability): *This metric allows you to monitor DB2 Universal Database for iSeries activity on your system. This metric applies to all systems running V4R5 or later and includes all database activity, including all SQL and data I/O operations. Use this metric to see how much of your system CPU is being consumed by a database function. By selecting a specific point on the system monitor graph, you can see which jobs are doing the most database activity. In addition, you can find detailed data for each job, including the number of milliseconds of CPU used by that job in database processing during the particular sample interval being graphed.*

The Database Capability metric measures how much processor time is being used for database activity. More specifically, the Database Capability shows database activity relative to the total database capability. Since the total database capability for the Dedicated Server for Domino is 15% CPU, the Database Capability metric would show 50%, for example, when 7.5% CPU on the Dedicated Server for Domino is being consumed by DB2 database processing. If the total DB2 database processing on the system was at the maximum 15% CPU, then the Database Capability metric would indicate 100% utilized.

Management Central provides dual thresholding function so you can set two limits when you want the system administrator notified that certain levels of CPU or capacity have been reached. Unique thresholds can be set for each metric being monitored such as Secondary Workload and Database Capability. For most of the metrics, Management Central provides the ability to select a point on the graph and display a list of the jobs that are consuming processing time for that metric. This can help identify the heavy users of resources such as DB2 database processing. You can drill down on individual the jobs in the list for additional information.

Additional techniques for observing the amount of DB2 database processing that is occurring are described in the following section. Unlike the Management Central Database Capability metric, these other techniques show the actual amount of CPU being used, rather than the relative terms reflected in Database Capability in Management Central.

Other methods for observing DB2 database CPU utilization

Several techniques are available for observing the amount of DB2 database processing on a Dedicated Server for Domino or a traditional server in addition to Operations Navigator Management Central as described above. The methods described below can be used to observe the amount of DB2 database CPU processing that is occurring on an iSeries or AS/400 with OS/400 V4R5 or later.

Work with System Status (WRKSYSSTS) command

This command is included with OS/400. On the Work with System Status display (Figure 12-2), the amount of DB2 database processing occurring can be observed via the *% DB capability* statistic.

Work with System Status						FROGGER	
						05/10/01	11:00:48
% CPU used : 24.0				Auxiliary storage:			
% DB capability : .0				System ASP : 310.3 G			
Elapsed time : 00:00:05				% system ASP used . . : 49.0248			
Jobs in system : 1293				Total : 326.4 G			
% perm addresses : .010				Current unprotect used : 1119 M			
% temp addresses : .012				Maximum unprotect . . : 1120 M			
Type changes (if allowed), press Enter.							
System	Pool	Reserved	Max	-----DB-----		---Non-DB---	
Pool	Size (M)	Size (M)	Active	Fault	Pages	Fault	Pages
1	935.60	588.51	+++++	.0	.0	.0	.0
2	11352.39	.28	2000	.0	.0	1.7	1.9

Figure 12-2 Work with System Status display

Work with System Activity (WRKSYSACT) command

This command is provided as part of the IBM Performance Tools for AS/400. On the Work with System Activity display (Figure 12-3), the amount of CPU being used by DB2 database is shown via the *Overall DB CPU util* statistic.

Work with System Activity							QBERT		
							05/10/01	10:59:20	
Automatic refresh in seconds							5		
Elapsed time			00:00:01		Average CPU util5		
Number of CPUs			2		Maximum CPU util9		
Overall DB CPU util . . .			: .0		Minimum CPU util1		
Type options, press Enter.									
1=Monitor job			5=Work with job						
							Total	Total	DB
	Job or					CPU	Sync	Async	CPU
Opt	Task	User	Number	Thread	Pty	Util	I/O	I/O	Util
	QPADEV0002	DJOHNSON	043837	00000007	1	.3	0	0	.0

Figure 12-3 Work with System Activity display

The IBM Performance Tools for iSeries

The Performance Tools can be used to monitor system performance over an extended period of time. These tools also provide the ability to summarize performance for desired periods of time using a variety of predefined reports. In the Workload section of the System Report, the *Total CPU Utilization (Database Capability)* statistic shows the amount of CPU processing that was spent in DB2 database. Figure 12-4 shows the Workload section of a sample System Report.

System Report									
Workload									
Member . . .	: PERFMON02	Model/Serial . .	: 270/10-4D2RM	Main storage . .	: 8192.0 M	Started	: 05/08/01 14:00:00	Page 0001	
Library . . .	: NQBE050801	System name . .	: QBERT	Version/Release :	4/ 5.0	Stopped	: 05/08/01 14:29:59		
Partition ID :	00	Feature Code . .	: 22AA-2253-1516						
QPFRADJ . . .	: 0	QDYNPTYSCD . .	: 1	QDYNPTYADJ . . .	: 1				
Interactive Workload									
Job Type	Number Transactions	Average Response	Logical DB I/O Count	----- Printer -----		Communications I/O Count	MRT Max Time		
				Lines	Pages				
There was no interactive data collected for the jobs you specified.									
Non-Interactive Workload									
Job Type	Number Of Jobs	Logical DB I/O Count	----- Printer -----		Communications I/O Count	CPU Per Logical I/O	Logical I/O /Second		
			Lines	Pages					
Batch	6	30	0	0	0	95.3338	.0		
Total	6	30	0	0	0				
Average						95.3338	.0		
Average CPU Utilization (Interactive Jobs) . . : 82.1									
CPU 1 Utilization : 79.4									
CPU 2 Utilization : 84.9									
Total CPU Utilization (Interactive Feature). . : .0									
Total CPU Utilization (Database Capability). . : .1									

Figure 12-4 Performance Tools System Report

Using these methods to observe the amount of DB2 database processing that is occurring can help determine whether a given application will fall within the 15% DB2 database processing guideline for a Dedicated Server.

12.5 Observing Domino processing with Performance Explorer

For those of you who might be interested in observing how much processing time is being spent specifically in Domino code (which would not include any OS/400 services or external programs that Domino might call), the Performance Explorer (PEX) can help you do exactly that. Prior to the V5R1 Dedicated Server for Domino enhancements, the process described below was used to determine whether an application was spending enough time in Domino processing to predict whether it would run well on a Dedicated Server for Domino.

Note, with the V5R1 enhancements of Fall 2001 described 12.2.1, "Enhancements with OS/400 V5R1" on page 343, the process described below is *no longer needed* since non-Domino processing is complementary when it is run in conjunction with Domino-based processing. Nonetheless, the information is provided here for those with an interest in lower level performance analysis.

Tools such as the Work with Active Jobs (WRKACTJOB) command provide a high level view of CPU consumption showing the amount of work running under the QNOTES user profile. However, when a Domino application calls system services as well as many other non-Domino functions, this non-Domino processing runs under the same job structure and QNOTES user profile.

Therefore, you *cannot* assume that all processing that runs using the QNOTES user profile is Domino processing. The best method for determining the exact amount of Domino processing is by using the Performance Explorer.

The OS/400 Performance Explorer tool provides detailed information on the use of CPU cycles at a library, job, program, or procedure level. Here are the recommended prerequisites for using the PEX tool for Dedicated Server application analysis:

- ▶ OS/400 at least at V4R3 (V4R4, V4R5, or V5R1 is preferred for a more accurate estimate)
- ▶ Domino 4.6 or higher (R5 is preferred for a more accurate estimate)
- ▶ iSeries or AS/400 Performance Tools Licensed Program

Notes:

- ▶ Installation of the Performance Tools is only required for printing PEX reports. PEX data can be collected without the Performance Tools product being installed.
- ▶ You can run PEX on any iSeries or AS/400 server that meets the above requirements. A Dedicated Server is not required for this analysis.

To use PEX to collect, report, and analyze Domino data on your iSeries or AS/400, follow these steps:

1. Use the following command to create a definition for the PEX data that you plan to collect:

```
ADDPEXDFN DFN(TPROFRC5) TYPE(*TRACE) JOB(*ALL) TASK(*ALL) MAXSTG(100000)
INTERVAL(5) TRCTYPE(*SLTEVT) SLTEVT(*YES) BASEVT(*PMCO)
TEXT('5 millisecond profile based on run cycles')
```

2. Use the following command to create a library for storing the data that the PEX tool collects:

```
CRTLIB LIB(DSDPEXDATA) TEXT('a place for my Dedicated Server PEX data')
```

Note: You can use any library name you choose. Just remember to replace DSDPEXDATA with the specific library name you have chosen in all of the remaining commands as well.

3. Before you begin collecting data, ensure that your Domino application or applications are running in a way that is representative of what you want to measure. In addition, *system CPU utilization* should be *above 50%* to ensure a high degree of confidence in the results.

4. Use the following command to begin the PEX data collection process:

```
STRPEX SSNID(younameit) DFN(TPROFRC5)
```

Note: If you want to measure multiple scenarios, simply use a different session name (the SSNID parameter) for each scenario. If you want to reuse a session ID, you must specify RPLDTA(*YES) on the ENDPEX command (Step 5) for the system to write over previously collected data.

5. Wait 20 to 30 minutes while PEX collects data. Then use the following command to end the data collection process:

```
ENDPEX SSNID(younameit) DTALIB(DSDPEXDATA) TEXT('brief description of test')
```

When the PEX data collection ends, the system deposits the data into the library you created in step 2.

6. To generate the PEX report, use the following command:

```
PRTPEXRPT MBR(younameit) LIB(DSDPEXDATA) TYPE(*PROFILE)
PROFILEOPT(*SAMPLECOUNT *PROCEDURE)
```


Note: If you do not have the Performance Tools LP installed on this iSeries or AS/400 server, you need to move the library with the data, DSDPEXDATA, to a system that has the Performance Tools LP.

To analyze the PEX report at a library level, start on page 3 of the report as shown in Figure 12-5. Look for the line that says QNOTES in the Name column. On this line, observe the value in the Hit % column. This value tells you what percent of the CPU cycles consumed on the server during your test was spent executing Domino code. In addition to the type of information shown below, the PEX report provides a list of all the jobs and tasks active on the system during the PEX collection. You can use this information to better understand how the various applications are using system resources.

Performance Explorer Report							5/09/01 05:34:20
Profile Information							Page 3
Library Section							
Library . . : DOMIN0507							
Member. . . : TPROF01							
Description : Web Shopper DB2 over DRDA to system Clara 78 users, 8 iterations							
Histogram	Hit Cnt	Hit %	Cum %	Start Addr	Map Flag	Stmt Numb	Name
	1080	2.0	19.6	FFFFFFFF8036CF84	MP	0	QU LIC
	1	0.0	19.6	263ED2E5D1004D34	MP	0	QNOTESINT
	34712	63.4	83.0	0743B72BD507B960	MP	0	QNOTES
	11	0.0	83.0	FFFFFFFFC10BE400	MP	0	QM LIC
	48	0.1	83.1	FFFFFFFFC1208E38	MP	0	PX LIC
	4	0.0	99.7	FFFFFFFFC1574E3C	MP	0	M2 LIC

Figure 12-5 Performance Explorer report: Library section

The Print PEX Report (PRTPEXRPT) command described earlier also generates a list of all of the procedures used by Domino, OS/400, and other applications that were active during the trace. This can help you to identify “hot spots” in code that may be candidates for optimization. An example of the procedure list is provided in Figure 12-6. This data was collected while a Mail and Calendar Users workload was active. All of the procedures are either Domino or OS/400, and only the first 20 entries have been included.

Performance Explorer Report							5/13/01 19:40:35
Profile Information							Page 16
Library . . : NQBE051301							
Member. . . : TPROF01							
Description : BLANK							
Histogram	Hit Cnt	Hit %	Cum %	Start Addr	Map Flag	Stmt Nbr	Name
*	28867	5.5	5.5	1196FA8E6933E2AC	==	0	LIBNOTES OSSEM/OSLockSem
*	20984	4.0	9.4	FFFFFFFFF0612B0	++	0	TDASMBL/switchin
*	20900	4.0	13.4	1196FA8E693A72AC	==	0	LIBNOTES VPPOOL/OSVBlockHandle
*	20651	3.9	17.3	1196FA8E693A50A0	==	0	LIBNOTES DPOOL/AllocDBlock
*	19521	3.7	21.0	1196FA8E693E37A4	==	0	LIBNOTES VARRAY/OSAddressInVARRA
*	15998	3.0	24.0	1196FA8E697288AC	==	0	LIBNOTES DBUNK/DbUNKGetName
*	15982	3.0	27.0	1196FA8E693ACEA0	==	0	LIBNOTES BPOOL/OSBBlockAddr
*	15309	2.9	29.9	1196FA8E693A5FA0	==	0	LIBNOTES DPOOL/FreeDBlock
*	14747	2.8	32.7	1196FA8E6933E0AC	==	0	LIBNOTES OSSEM/OSUnlockSem
	14045	2.7	35.4	1196FA8E693543CC	==	0	LIBNOTES MEMORY/LockMemHandle
	12050	2.3	37.6	FFFFFFFFF074AFC	++	0	STRNGEAO/memmovwteaofree
	11971	2.3	39.9	FFFFFFFFF800C00	++	0	CFPLBLA/bla_Pulsar
	11333	2.1	42.0	1196FA8E693408EC	==	0	LIBNOTES OSSEM/OSUnlockSpin
	9285	1.8	43.8	1196FA8E699E23C0	==	0	LIBNOTES OPEN/UnlockCollection
	8936	1.7	45.5	1196FA8E693296AC	==	0	LIBNOTES MEMINIT/AccessSHTChunks
	8175	1.5	47.0	1196FA8E69350ECC	==	0	LIBNOTES MEMLOCK/LockHandle
	6413	1.2	48.3	1196FA8E69353A0C	==	0	LIBNOTES MEMORY/OSMemoryLock
	6240	1.2	49.4	1196FA8E697290CC	==	0	LIBNOTES DBUNK/GetUNKAddr
	6125	1.2	50.6	1196FA8E6972882C	==	0	LIBNOTES DBUNK/

Figure 12-6 Performance Explorer report: Profile information

You can find more detailed information on PEX in 4.14, “Performance Explorer (PEX)” on page 116, in the redbook *AS/400 Performance Explorer Tips and Techniques*, SG24-4781, and at the Web site: <http://www-1.ibm.com/servers/eserver/series/sftsol/pex.htm>

12.6 Sizing a Dedicated Server for your Domino application

The recommended method for sizing Domino workloads is to use the IBM Workload Estimator for iSeries. See Chapter 3, “Sizing Domino for iSeries using the Workload Estimator” on page 19, for an in-depth discussion. You can access the Workload Estimator from the Domino on iSeries home page (select **Sizing information**) or at: <http://as400service.ibm.com/estimator>

The Workload Estimator is refreshed on a regular basis. At the time this redbook was written, the two latest refreshes are April 2001 and June 2001. Several Domino-related enhancements of special interest when projecting workloads for the Dedicated Servers have been added. These include:

- The Estimator now supports projecting other workloads besides Domino on the Dedicated Server for Domino. The Estimator allows the combined effect of Domino, Java, Net.Commerce, WebSphere, and traditional workloads to be simultaneously estimated for a single server.

Note: The Workload Estimator requires that Domino comprise at least 50% of the projected CPU utilization when projecting for a Dedicated Server.

- A projection of DB2 database processing is provided for Domino workloads as well as other workload types such as Java, WebSphere, and HTTP.

Here are the April 2001 and June 2001 Domino-related Workload Estimator enhancements as described in the Estimator's "What's New" help text:

- **Domino enhancements:** The Domino workload has been enhanced to include new defaults and assumptions based on feedback received from sales representatives and business partners about how typical customers are using Domino as well as how they are using the Workload Estimator to project Domino workloads. Along with these enhancements, a couple of user interface adjustments have been made to eliminate some ambiguity in both Domino planned usage, as well as classifying the complexity of a Domino user written application. Even if you are familiar with previous versions of the Workload Estimator, we strongly suggest that you visit the Domino workload help text for a more complete explanation of the Domino workload enhancements.
- **New DSD calculations:** New with OS/400 V5R1, the Dedicated Server for Domino models will support Domino complementary workloads such as Java servlets and WebSphere Application Server. This includes the five new Dedicated Server for Domino models announced April 2001, as well as previous Dedicated Server for Domino models. When specifying V5R1 for the operating system version, the Estimator will allow other workloads in addition to Domino on a Dedicated Server for Domino as long as the following requirements are met:
 - Domino must be the majority of the projected workload.
 - The projected DB2 Universal Database utilization is less than the rated DB2 capacity for that model.

The rated DB2 capacity is 15% of the CPU for Dedicated Server for Domino models. The new V5R1 function that supports complementary workloads for Dedicated Server for Domino models has been available since 28 September 2001 with the refreshed version of OS/400 V5R1. Prior to that date, you may want to specify V4R5 for OS Version when projecting Domino workloads for previously installed Dedicated Server for Domino systems. Removal of Domino 4.6 Support – Domino Version 4.6 will no longer be supported.

Note: All Domino Version 4.6 workload definitions previously sized and saved will automatically be converted to Domino Version 5 when restored into the Workload Estimator.

- **iNotes Outlook Mail Client added to the Domino workload:** The Domino workload has been enhanced to include the iNotes Outlook Mail client as an option.
- **Lotus QuickPlace:** The help text for the Domino workload has been updated to include the instructions needed to size a QuickPlace installation. Visit the Domino workload help text for a more complete explanation.

12.6.1 Special tuning for the Dedicated Server

No special tuning is required for a Dedicated Server. General recommendations for improving the performance of Domino on traditional iSeries models apply to the Dedicated Server as well.

We encourage you to use the latest supported version of Domino for iSeries and to regularly apply the maintenance updates and OS/400 PTFs. The Lotus Domino site <http://www.iseries.ibm.com/domino/support/> provides information specific to Lotus Domino on AS/400 and iSeries for:

- ▶ Maintenance Release information (formerly called QMRs)
- ▶ Maintenance Updates (MUs)
- ▶ Other helpful support information



Domino transaction logging on iSeries

Introduced with Domino R5 for all platforms, transaction logging (TxL) was described in the Release Notes. It states that “*transaction logging* is an industry-standard technique for reliable data storage. A transaction log is a sequential file to which everybody writes; sequential writing on disk is much faster than writing in various places on disk.”

For anyone who knows how the iSeries architecture provides automatic storage management and optimization through its implementation of single-level storage, this statement immediately raises the question of whether TxL may impact performance on an iSeries server.

This chapter describes the advantages and disadvantages of Domino transaction logging from an iSeries perspective. First it gives a brief overview of iSeries storage management followed by an introduction of Domino transaction logging. Then it presents some results of performance measurements, along with conclusions and recommendations.

13.1 Storage management on iSeries

On an iSeries server, main memory is called *main storage*. Disk storage is called *auxiliary storage*. You may also hear disk storage referred to as *direct access storage device (DASD)*.

Many other computer systems require you to take responsibility for how information is stored on disks. When you create a new file, you must tell the system where to put the file and how big to make it. You must balance files across different disk units to provide good system performance. If you discover later that a file needs to be larger, you need to copy it to a location on a disk that has enough space for the new, larger file. You may need to move other files to maintain system performance.

The iSeries server takes responsibility for managing the information in auxiliary storage. When you create a stream file (for example a Domino database), an SQL table or any other OS/400 object type, the system places it in the best location for good performance. In fact, it may spread the data in the file across multiple disk units. When you add more records to the file, the system assigns additional space on one or more disk units.

The system uses a function that is called *virtual storage* to create a logical picture of how the data looks. This logical picture is similar to the way we think of the data. In virtual storage, all of the records that are in a file are together (contiguous), even though they may be physically spread across multiple disk units in auxiliary storage.

Single-level storage is the unique architecture of the iSeries server that allows main storage, auxiliary storage, and virtual storage to work together accurately and efficiently. With single-level storage, programs and system users ask for data by name, not by where the data is located.

13.1.1 Single-level storage

Application programs on an iSeries or AS/400e server are unaware of the underlying hardware characteristics, because of the *Technology-Independent Machine Interface (TIMI)*. Applications are also unaware of the characteristics of any storage devices on the system because of single-level storage.

As with TIMI, the concept of single-level storage means that the knowledge of the underlying characteristics of hardware devices (in this case, main storage and disk storage) reside in the *System Licensed Internal Code (SLIC)*. All of the storage is automatically managed by the system. No user intervention is ever needed to take full advantage of any storage technology. Programs work with objects. Objects are accessed by name, not by address.

The iSeries and AS/400e server address space is vast. iSeries and AS/400e models can address the number of bytes that 64 bits allows it to address. The value 2^{64} is equal to 18,446,744,073,709,551,616. Therefore, the iSeries and AS/400e models can address 18.4 quintillion bytes. To put this into more meaningful terms, it is twice the number of millimeters in a light year. Light travels at approximately 6,000,000,000,000 miles in one year. Refer to the *IBM @server iSeries Handbook*, G19-5486, for more information on iSeries architecture.

13.1.2 When data is actually written to disk

The virtual storage function keeps track of where the most current copy of any piece of information is – in main storage or in auxiliary storage. This also enables another extremely important iSeries and AS/400 benefit – disk I/O optimization. SLIC takes care of:

- ▶ Where on a particular disk (as a part of all auxiliary storage) each piece of information will be stored, when an object is created.
- ▶ When changed data is actually moved from main storage to disk.

For example, normally the system determines when changed records of a DB2 file are written to auxiliary storage, unless the user specified the *Records to force a write (FRCRATIO)* parameter when creating the physical file with Create Physical File (CRTPF) command.

Domino databases are not stored in physical files. They are rather implemented as stream files in the Integrated File System (IFS). However, when stream files are changed by an application, such as Domino, some functions within the microcode (SLIC) decide when and how often changed data will be written to disk. This can optimize the number of physical I/O operations. The “sync daemon” P0FSYNC is responsible for forcing stream file changes to disk, if that was not already done by storage management within a certain time limit.

In the context of this chapter as a consequence of the two facts mentioned above, we can draw the following conclusions:

- ▶ By default, the user does not select a specific disk drive when storing data in iSeries auxiliary storage. This applies as well for the contents of the Domino data directory and other directories used by Domino. However, there is a way to do so described in 13.1.3, “Auxiliary storage pools” on page 373.
- ▶ If an iSeries application, such as Domino for iSeries, implements its own scheme to defer and group write operations to disk drives, you may see less or no performance improvements compared to other operating systems with less sophisticated storage management.

The latter statement is the reason why we are skeptical about performance improvements caused by the implementation, but this is discussed in more depth later in this chapter.

13.1.3 Auxiliary storage pools

An *auxiliary storage pool (ASP)*, also referred to as a disk pool in Operations Navigator, is a software definition of a group of disk units on your system. Conceptually, each ASP on your system is a separate pool of disk units for single-level storage. The system spreads data evenly across the disk units within an ASP. There are three types of ASPs:

- ▶ System auxiliary storage pool
- ▶ User auxiliary storage pools
- ▶ Independent auxiliary storage pools

Your iSeries server may have many disk units attached to it for ASP storage. To your system, they look like a single unit of storage. You can use ASPs to separate your disk units into logical subsets. For more ideas on how to use auxiliary storage pools on your system, see the iSeries Information Center at <http://publib.boulder.ibm.com/html/as400/> and search for “ASP”.

When you have multiple ASPs configured on your system, each ASP can have different strategies for availability, backup and recovery, and performance. The availability and backup and recovery-related issues are now more or less topics of the past since they can be addressed much better by using *Redundant Array of Independent Disks Type 5 (RAID5)* or *mirrored storage protection (RAID0)*.

Using an ASP to improve disk performance

What is more interesting in the context of Domino transaction logging is the performance improvement that can be gained, if ASPs are wisely used. In situations where an extremely high amount of data permanently has to be written sequentially to a single object, the disk response can be dramatically reduced by placing that object on a dedicated disk unit. Or more specifically, the data must be placed under one (or more) disk arms, which are not used to access any other information.

This method has always been recommended for highly active OS/400 journal receivers. Although *not* implemented through OS/400 journal objects, the concept of Domino transaction logging is very similar to DB2 UDB for iSeries journaling, which has been implemented on IBM System/38 over 20 years ago and since then successfully used on AS/400 and iSeries servers.

For more information on improving disk performance with ASPs, see the iSeries Information Center at <http://publib.boulder.ibm.com/html/as400/> and search for the string “Using ASPs for improved performance”.

In other words, according to Technote 179858, the recommendation for other Domino platforms sounds like this:

“The transactional logs must be on a separate physical drive for there to be any performance improvement.” See 13.2.1, “Performance improvement due to Domino transaction logging” on page 376, for more details.

On iSeries, the statement translates to:

“The Domino transactional logs should be placed into a separate user ASP to avoid performance degradation due to the additional disk I/O operations.”

To place a directory to be used by Domino for iSeries into a user ASP, you create and mount an OS/400 *user-defined file system (UDFS)*. While an integrated file system directory can be created only in the system ASP, a UDFS and its directory structure can be created in any available ASP.

Creating a new auxiliary storage pool

After you install new disks, you can create a new ASP with the Dedicated System Tools (DST) menu or Operations Navigator. If you want to move an existing disk drive from one pool to another, you need to use DST.

To change the size of an ASP, complete the following steps:

1. From the IPL or Install the System display, select option 3 (Use Dedicated Service Tools). Sign on using the QSECOFR profile.
2. From the DST menu, select option 4 (Work with disk units).
3. Select option 1 (Work with disk configuration).
4. Select option 3 (Work with ASP configuration).
5. From the Work with ASP Configuration menu, select the option that matches your task and follow the screen prompts to complete the task.

You can find more information about working with an ASP in *OS/400 Backup and Recovery*, SC41-5304. Or go to the iSeries Information Center at <http://as400bks.rochester.ibm.com/pubs/html/as400/v5r1/ic2924/index.htm> and look under **System planning and installation-> Hardware and software-> System hardware-> Managing disk units in disk pools-> Managing data in independent disk pools**. You may also download a PDF version of the book *Managing data in independent disk pools* from this site by selecting **System planning and installation-> Hardware and software-> System hardware-> Print this topics**.

Creating and mounting a user-defined file system

Assume, you already created a new user ASP called ASP02, containing no data. This section describes a sample procedure to create a UDFS in ASP02 and mount it for use. Two views are presented:

- ▶ Operations Navigator (recommended)
- ▶ Command Language (CL) equivalent for those who prefer to use a 5250 session; see “Creating and mounting a UDFS using OS/400 CL commands” on page 376

For a more detailed description and examples, see Chapter 10 of the redbook *Lotus Domino for AS/400 Internet Mail and More*, SG24-5990.

To perform this task using Operation Navigator, follow these steps:

1. To create a UDFS on your iSeries server, open Operations Navigator and expand **File Systems-> Integrated File System-> Root-> dev**.

Under the dev folder, QASPnn folders correspond to each ASP that exists on the system. The QASPnn directories are automatically created when ASPs are added to the system.

2. To create a new UDFS object in *ASP02*, right-click **QASP02**. On the pop-up menu that appears, select **New UDFS**.
3. In the New User-Defined File System window, enter the name of the new UDFS and its description.

Once created, the newly created UDFS object appears on the right side of the Operations Navigator window.

4. To create an integrated file system directory onto which the UDFS object will be mounted, complete these tasks:
 - a. Right-click the **Root** file system, and select **New Folder**.
 - b. Specify `\domino\dommail1\logdir` as the name of the new integrated file system directory.
 - c. Click **OK** to create a new integrated file system directory named *logdir* within the existing Domino server data directory `\domino\dommail1`. This directory is used to map the file system in the user ASP to the integrated file system root file system.
5. Right-click the newly created UDFS object (**myudfs.udfs**). Select **Mount** in the pop-up menu that appears.

In the Mount User-Defined File System window that appears, specify the integrated file system directory path on which to mount the UDFS (it is `\domino\dommail1\logdir` in our example). Alternatively, click the **Browse** button to bring up a graphical directory tree of the system, and select the integrated file system directory of your choice. Then select the access type as **read only** or **read/write**. Click **OK** to finish.

As long as the UDFS object is mounted, any stream files placed in the integrated file system directory `\domino\dommail1\logdir` are stored in the ASP02 disk space of the `\dev\qasp02\myudfs.udfs` object.

Creating and mounting a UDFS using OS/400 CL commands

For those who prefer using OS/400 CL commands, the steps necessary to create a subset of the Domino data directory in an user ASP are outlined below.

A CL command equivalent to step 3 on page 375 uses the Create User-Defined FS (CRTUDFS) command. Enter the following command on any OS/400 command line:

```
CRTUDFS '\dev\qasp02\myudfs.udfs' TEXT('My UDFS in ASP 2')
```

To create a new directory use the following Make Directory (MD) CL command:

```
MD '/domino/dommail1/logdir'
```

The new file system can be mounted by entering the following command on any OS/400 command line:

```
MOUNT *udfs '\dev\qasp02\myudfs.udfs' '\domino\dommail1\moredbs'
```

As long as the UDFS object is mounted, any stream files placed in the integrated file system directory \domino\dommail1\logdir are stored in the ASP02 disk space of the \dev\qasp02\myudfs.udfs object.

For further information on user defined file systems, refer to the Information Center CD, SK3T-2027, shipped with your iSeries server or the iSeries Information Center at:
<http://www.iseries.ibm.com/infocenter>

For more information regarding UDFS with Lotus Domino for iSeries, see Chapter 10 in the redbook *Lotus Domino for AS/400 Internet Mail and More*, SG24-5990.

13.2 Domino transaction logging and how it operates

Domino transaction logging is a new feature in Domino R5. It is essentially a new method of writing out database changes to improve performance and to ensure data integrity. Most of the information in this section has been taken from Technote 179858 and modified regarding the unique architecture of iSeries servers.

A *transactional log* is simply a binary file where transactions are written. The transactions are saved in log extents that have a TXN extension. Each log extent is 64 MB and will fill before a new extent is created or a spare one is reused. Multiple log extents collectively can grow to a maximum size of 4 GB with circular logging or available disk space for archive logging.

Attention: Although serving a similar purpose, OS/400 journal objects and Domino transactional logs are *not* the same and are implemented differently. While journals are implemented as special OS/400 object types, the Domino transactional logs are stored in stream files within the OS/400 Integrated File System (IFS).

13.2.1 Performance improvement due to Domino transaction logging

According to Technote 179858, the main purpose of Domino transaction logging is to improve performance on the R5 server through sequential writes to the transactional logs.

When transaction logging is enabled on the server, the performance improvement is mainly due to the nature of how transaction logging operates. The write operations to the transactional log are sequential. This is faster since there is less head movement, provided the log is on a separate disk, is not used for other I/O, and there is never a need to search for a place on the disk to write as there is in R4 or if transaction logging is not enabled.

However, the above statement only holds true for disk constrained environments and for operating systems that do not provide functions to optimize disk access like OS/400 does as described in 13.1.1, “Single-level storage” on page 372.

Using separate disk drives

Technote 179858 also recommends that you place the transactional logs on a separate physical drive for there to be any performance improvement. It is not sufficient to simply redirect the logs to a separate partition or a separate logical drive. In general, if the transactional logs are on a separate drive, a 10 to 20% improvement should be seen. However, if the logs are put on the same drive, it is likely that there will be approximately a 60% degradation.

In OS/400 terms (see also “Using an ASP to improve disk performance” on page 374), this means you should place the transactional log for Domino into a user ASP, which does not contain any other data, to avoid the performance degradation due to the additional disk writes. See “Creating a new auxiliary storage pool” on page 374 for information on how to make a directory in a user ASP accessible for Domino.

Without using transaction logging and in Domino R4, writing to disk was time consuming under operating systems not providing single-level storage. Modifications could occur across multiple databases or different parts of one database. As a result, the head had to move over various areas of disk to change or update data. This means there was a significant amount of transaction time committing data to the actual NSF file. Again, this does not necessarily apply to an iSeries server, since all data is spread evenly over all disk drives (in the same ASP) and asynchronous writes can be performed in parallel.

When transaction logging is enabled, complete transactions are “committed” to the transactional log. All writes are done to the transactional log before they are ever written to the database. The writes are done sequentially at least after each transaction so the transactional log is up to date generally to the hundredth of a second. Again, since the writes are sequential, there is less I/O and performance is improved.

Flushing and hardening

Once changes are put into the transactional log, the changes must also eventually be “hardened” to the database, that is physically written to disk. This occurs through a process called *flushing*. Any open database has an in-memory version of the database that is held in the *Unified Buffer Manager (UBM)*. Flushing moves all changes that were made to the database but only kept in memory in the UBM, to the actual NSF file.

There is no set interval for this since the UBM determines when flushing will occur. It is usually done when there is a lull in the server activity. The *Database Instance ID (DBIID)* is used to correlate the updates in the transactional logs and in memory to the respective database. It is important to note, however, that the transactional logs are not read from during this process since the transactional logs are mainly a write-only object. The updates are read and flushed from the UBM. They are only read from the transactional logs during crash recovery.

Controlling performance: Favor Run Time or Restart Recover Time

The *Runtime/Restart Performance* field in the Server document (within the Domino directory Names.nsf, see “Setting up transaction logging” on page 379) determines how many MB of changes are kept in memory. The amount of space used is defined by the *performance level* chosen in the Server document. There are three choices:

- ▶ Standard (default)
- ▶ Favor Runtime
- ▶ Favor Restart Recovery Time

If *Standard* is selected, the Redo-Limit is 49 MB. This means that checkpoints during runtime are minimal, and 49 MB worth of changes are held in the UBM before they are flushed and hardened to databases.

The *Favor Runtime* choice has a Redo-Limit on the smaller side of 500 MB or half the defined log size. This means that more information is held in the UBM and hardened to the database less frequently. Allowing UBM to manage more of the database changes allows larger machines to better utilize memory and reduce unnecessary flushing.

The *Favor Restart Recovery Time* choice allows for more checkpoints during runtime. Less information is held in the UBM, and data is hardened to databases more frequently. The trade-off is that production time is slower but server restart is faster.

Checkpoints are internal state snap shots of all current activities. They are written into the transactional log to record the current state of all databases and transactions in the system at that moment in time. On restart, TxL first locates the last checkpoint written to the logs, then starts the recovery based on that state snap-shot and works forward. That is, the more frequently a checkpoint is written to the TxL, the sooner it can be located and the sooner recovery commences. The Runtime/restart Performance field is also used to select an appropriate regime of setting checkpoints to suit the performance style chosen. Table 13-1 shows both checkpoint intervals and cache/redo limits.

Table 13-1 Runtime performance cache redo and checkpoint frequencies

Performance style	Cache/redo max buffer size	Checkpoint frequency
1 = Favor runtime	500 Mb	500 Mb
2 = Standard (default)	49 Mb	16 Mb
3 = Favor Restart	15 Mb	5 Mb

Crash recovery

When transaction logging writes transactions to the logs, an UNDO record and a REDO record are usually committed for each transaction. First an UNDO log record is generated to be used in the event of a system outage. This is done before a change is written to a database.

Before committing a transaction, a REDO record is also generated. It is used to re-apply a transaction from the transactional log to the database in the event that it was not flushed to the NSF before a server outage. Undo and redo records ensure that if a change is half done, it will be fully undone, and if a change was completely done, then it will be fully redone to the NSF.

After a server outage, the transactional logs are played back. The Recovery Point is determined for each NSF requiring log updates; it is the oldest log information that needs to be re-applied to databases. The databases are restored to the exact moment of the outage, guaranteed to restore any data from a completed transaction. Partial transactions are undone and rolled back to the last good state in an effort to avoid corruption in the database. The partial work will be removed from the database before restart completes and the database is made available for use.

Note, view indexes are not transactionally logged in R5 so views may need rebuilding after a restart or media recovery. Attachments, however, are transactionally logged. It is important to note that attachments are logged as redo only, so if the database is recovered using media recovery, the last copy of the attachment (once they are done, they stay done) will be returned to you. If, however, the server crashes with uncommitted attachment updates, they will not be undone since an undo record is never created for them.

Without the benefit of transaction logging, the Domino FIXUP task relies on the fact that 99.9% of the data is present in the NSF to correct integrity problems.

You should not move databases away from a server and copy them over from another server after a server crash. The database may be missing a significant amount of changes that are only stored in the transaction log and if the database is not found during restart the changes will not be restored. Recovery restart must be performed first and can be triggered, for example, by a request to compact a nonexistent database.

Setting up transaction logging

To set up transaction logging, follow these steps:

1. Make sure that all databases you want to log are in the Domino data directory.
2. From the Domino Administrator, click the **Configuration** tab.
3. In the Use Directory on field, choose the **server's Domino Directory**.
4. Click **Server Configuration**, and then click **Current Server Document**.
5. Click the **Transactional Logging** tab.
6. Complete these fields as shown in Figure 13-1, and then save the document.

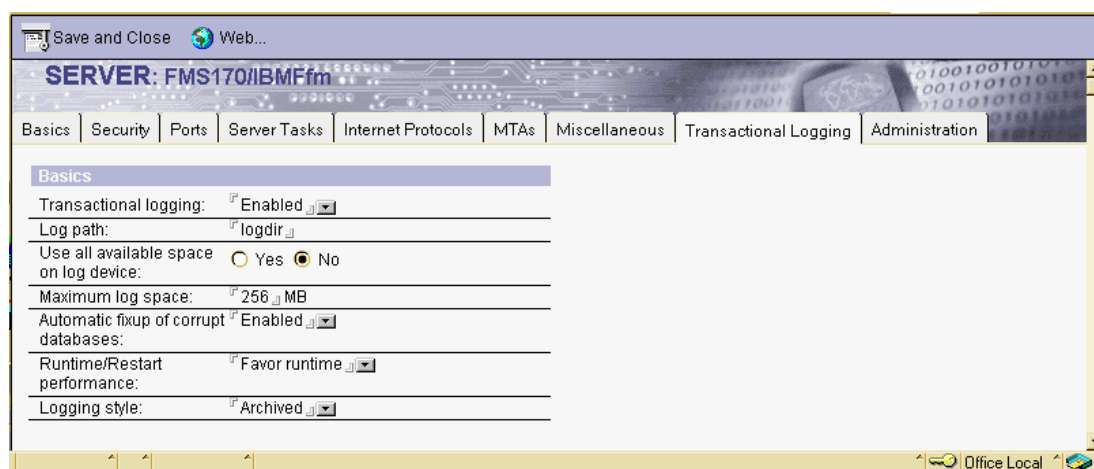


Figure 13-1 Enabling and configuring transaction logging in the server document

The server needs to be restarted to enable or disable transaction logging or to activate the configuration settings. The following section describes the parameters in detail.

Configuration parameters related to transaction logging

Transaction logging is enabled in the Server document. All the fields in the Server document map to specific notes.ini parameters.

Important: The TRANSLOG_ parameters are always overwritten by the values from the server document, so there is no point to change them in notes.ini directly. Use the method shown in Figure 13-1 instead.

The parameters are:

- **TRANSLOG_Status:** Whether transaction logging is Enabled (1) or Disabled (0). The default is Disabled; choose Enabled.

- **TRANSLOG_Path:** Specify the path to the TXN files.

The default path name is LOGDIR in the Domino data directory. We strongly recommend that you store the log on a separate, mirrored ASP with a dedicated controller (Disk IOP). See “Creating a new auxiliary storage pool” on page 374 for information on how to make a directory in a user ASP accessible for Domino.

In theory, you could also use RAID5 protected disks, but this makes only sense in an environment where you need more than one disk drive to contain the transactional log, which is very unlikely with the capacity of currently available disk drives. However, if you decide to use RAID5 or a disk controller not dedicated to this ASP, you should closely monitor the utilization of the disk IOP as described in 4.11, “Collecting performance data” on page 83 and “Disk IOP utilization” on page 93.

The ASP containing the directory should have at least 1 GB of disk space for the transaction log. If you are using the device solely for storing the transaction log, set the “Use all available space on log device” field to Yes.

- **TRANSLOG_UseAll:** Specify whether you want to use all available space. If enabled, you do not need to enter a value in the “Maximum log space” field.
- **TRANSLOG_MaxSize:** If you placed the log on a separate disk, specify as much space as is available. Values over 200 MB are recommended. The maximum size, in MB, is for the transaction log. The default is 192 MB; the maximum is 4096 MB (4 GB). Domino formats at least three and up to 64 log files, depending on the maximum log space you allocate. It is used only for Circular style logging.
- **TRANSLOG_AutoFixup:** Enabled by default. This means that, if a database is found to be corrupt after restart, in which case the transaction log cannot be used to recover it, Domino runs the Fixup task automatically. If disabled, Domino notifies the administrator to retrieve a backup and recover the database from the logs, or to run Fixup manually.
- **TRANSLOG_Performance:** Favor runtime (1), Standard (2), Favor restart recovery (3). This parameter allows you to control how often Domino records a recovery checkpoint in the transaction log. Checkpoints entail system overhead, but they are important for restart performance. You can choose the default (recommended), or skew the frequency of checkpoints to favor server performance (fewer checkpoints) or to shorten recovery time (more checkpoints). The fewer the checkpoints there are, the longer restart will take if a system failure occurs. See “Controlling performance: Favor Run Time or Restart Recover Time” on page 377 for a more in-depth discussion.
- **TRANSLOG_Style:** Circular (0) versus Archive (1).

Choose Circular (default) to continuously re-use the log files and overwrite old transactions. You are limited to restoring only the transactions stored in the transaction log.

Choose Archive (recommended) to not re-use the log files until they are archived. A log file can be archived when it is inactive, which means that it does not contain any transactions necessary for a restart recovery. Use a third-party backup utility to copy and archive the existing log. When Domino starts using the existing file again, it increments the log file name. If all the log files become inactive and are not archived, Domino creates additional log files.

Choose Circular (the default) only if you do not have an R5 compatible backup utility in place.

Important: If you want to use Domino for iSeries online backup provided by Backup Recovery and Media Services (BRMS), you have to change this parameter to Archive.

After you understood the basics about how OS/400 handles disk I/O operations in 13.1, “Storage management on iSeries” on page 372, and the principles of Domino transaction logging in this section, you are ready to interpret some testing results.

13.3 Performance tests with TxL on Domino for iSeries

Based on numerous questions regarding the performance impacts of Domino transactional logging on iSeries and AS/400 servers, the IBM Rochester performance group conducted several tests to explore its performance behavior in a controlled environment. All tests used a workload similar to the NotesBench R5Mail and were performed on an iSeries Model 820 server. It is expected that the results and conclusions can generally be applied to other iSeries and AS/400 servers.

13.3.1 Test environment

The main objective of the tests described below was to study the impact of Domino transaction logging on the responses times for Notes clients, the CPU, and disk utilization in different scenarios:

- ▶ **Scenario 1:** No transaction logging
- ▶ **Scenario 2:** TxL *dedicated* user ASP
- ▶ **Scenario 3:** TxL *non-dedicated* in same ASP as the Domino data directory
- ▶ **Scenario 4:** TxL dedicated user ASP with “*Favor Recovery Time*”
- ▶ **Scenario 5:** TxL dedicated user ASP with *less memory*
- ▶ **Scenario 6:** TxL *non-dedicated* ASP with *less memory*
- ▶ **Scenario 7:** TxL dedicated user ASP with less memory *limited disk space for the transaction log*

These tests allow the following comparisons to be considered:

- ▶ Logging versus no logging
- ▶ Logging to the same versus different drives as the Domino mail databases
- ▶ Favoring runtime versus favor restart recovery time logging option
- ▶ Use all available disk space versus limiting the disk space to 1 GB
- ▶ Memory rich versus a less memory rich (but not constrained) environment

For all of the tests, the Domino mail databases used by the workload resided in ASP1, the system ASP. When testing the dedicated ASP for the transaction log, we used ASP3 which was configured with two mirrored drives. For these tests a User Defined File System (UDFS) was created which was defined to reside on ASP3 (as described in “Creating and mounting a UDFS using OS/400 CL commands” on page 376).

All tests were conducted by simulating a constant workload produced by 6,000 users performing in average the following activities every 15 minutes:

- ▶ Read five documents
- ▶ Update two documents
- ▶ Delete two documents
- ▶ Scroll a view once
- ▶ Open and close one database
- ▶ Open and close one view
- ▶ Send one memo to three recipients
- ▶ Do three lookups on the Domino Directory

Every 90 minutes, the test schedules one appointment and sends one invitation to the recipients. There are also server name lookups, and messages are deposited in the mail databases.

13.3.2 Test results

The main results of the tests are summarized in Table 13-2.

Table 13-2 System level results of TxL performance tests

	Ded. disk	Favor	Mem GB	TxL disk space	CPU %	Resp. time	Disk I/O/sec		Disk Util. %	
							ASP1	ASP3	ASP1	ASP3
1	-	-	12	-	38.0	32ms	267.0	-	7.7	-
2	Y	Run	12	All	42.3	42ms	269.6	50.6	11.4	8.9
3	N	Run	12	All	41.9	74ms	329.8	-	19.2	-
4	Y	Restart	12	All	46.2	85ms	379.3	52.7	10.3	9.5
5	Y	Run	8	All	44.2	62ms	388.6	51.9	18.5	9.6
6	N	Run	8	All	43.6	118ms	433.2	-	25.2	-
7	Y	Run	8	1GB	42.6	60ms	384.4	51.3	17.8	9.6

Note, the disk I/O per second reported here is from the performance monitor reports. The reports show the I/O counts that are sent the disk controller. For RAID5 configurations, one write I/O generates four physical I/O requests: two reads and two writes. Depending on the disk controller write cache, each disk device's read ahead buffer, and other I/O requests to the device, these I/O requests may be grouped with others and a single larger I/O to the device may actually satisfy multiple smaller requests.

The Average Disk Utilization reflects the actual utilization of the disk devices. An environment with a larger percentage of write I/O per second will generate higher disk utilizations than one with a larger percentage of reads.

13.3.3 Conclusions

Several conclusions can be drawn from the test results shown above.

Performance improvements

Implementing transactional logging on iSeries should be done for recovery reasons since it does not improve performance as reported for some other operating systems. Transactional logging, in all cases tested, increased system CPU utilization, increased disk I/O per second, and caused higher response times. The extent of the increases depends on how the system is configured and which transactional logging options are selected.

So why have many claimed that transactional logging will improve performance? Well, there are a couple of reasons for this. Early in Domino Version 5, it was widely held that enabling transactional logging improved performance. This was primarily because the transactional logging testing was being done in disk constrained environments. And for other operating systems, they may in fact see performance improvement because enabling transactional logging makes more of the I/O asynchronous, which iSeries always did automatically. The term *asynchronous I/O* means the application does not have to wait for all physical disk operations to complete.

Meanwhile other operating systems have also reported that transactional logging may not necessarily improve performance unless they are in a disk constrained environment. iSeries sizing and capacity planning tools, such as the Workload Estimator (see Chapter 3, “Sizing Domino for iSeries using the Workload Estimator” on page 19), are designed to configure balanced systems that are not disk constrained. In addition, the single level store, as described in 13.1.2, “When data is actually written to disk” on page 373, and I/O subsystem architecture optimize all I/O activity by using disk and controller caches. In fact, they do this best when applications do not attempt to manage the I/O processing themselves.

For example, in scenario 1 (no logging), 190 (71%) of the 267 I/Os per seconds were asynchronous. In scenario 2 (logging to a dedicated ASP), 201 (63%) of the 320 I/Os per second were asynchronous. The majority of the increase in I/O per second in scenario 2 was actually synchronous I/O. Since applications, in this case a Domino task, need to wait until synchronous I/O operations are completed, this may be the main reason for increased response times.

Using dedicated disk arms

When implementing transactional logging on iSeries, you should place the transaction logs on their own drives to limit impact on response time performance. This is the same recommendation as for other operating systems and what we said in “Using separate disk drives” on page 377.

Comparing scenario 2 (log to ASP3) and scenario 3 (log to ASP1), we see that logging to the same drives as the Domino mail databases significantly degraded response time, even though CPU utilization and disk I/O per second were about the same. Logging to the same drives as the mail databases caused higher contention for the disk drives, which increased user response times by about 75% from 42 to 74 milliseconds.

A similar comparison can be made between scenarios 5 and 6 for a less memory rich environment. In this comparison, the system CPU and disk I/O per second was again approximately the same for the two scenarios, but logging to the same drives as the mail databases caused response time to increase from 62 to 118 milliseconds, or 90%.

Optimizing for run time or restart time

Favor Runtime performance provides better run time performance than Favor Restart Recovery Time. Comparing scenario 2 (Favor Run Time) and scenario 4 (Favor Restart Recover Time), Favor Restart Recovery Time nearly doubled the response time, generated 25% more total system I/O and increased CPU by about 9%. In scenario 4, the Domino server was doing more forcing of the mail databases itself rather than allowing the system to optimize the writing out of the changed mail databases to disk.

You can set this option to control the behavior of transaction logging in the server document as described in “Controlling performance: Favor Run Time or Restart Recover Time” on page 377 and “Configuration parameters related to transaction logging” on page 379.

Limiting the disk space to be used

In scenario 7, we configured transactional logging to log to ASP3 and to not use all available disk space, but rather to only use 1 GB worth of space. As expected, the results show essentially no difference from scenario 5 in which the logs used closer to 4 GB worth of disk space. Limiting the log to using 1 GB may be a better option if the logs are going to reside on the same drives as your applications and databases, and it does limit the number of log entries before the log wraps on itself when using circular logging.

13.4 When to use Domino for iSeries transaction logging

As we showed with the test environment described in 13.3.1, “Test environment” on page 381, there are not necessarily any run-time performance benefits for using Domino transaction logging, simply because the iSeries server is already fully I/O optimized and adding TxL will impose a small overhead because by definition, the data is being written twice instead of once.

You will, however, achieve huge improvements in restart times should Domino ever crash. Your data integrity and exposure to lost records after a crash is vastly reduced compared to using FIXUP task, which may be necessary to run.

Another good reason to use transaction logging is reductions in backup data volumes and timings by up to 90% through *incremental backups* on all Domino databases including Mail, Domino.Doc, and QuickPlace.

13.4.1 Incremental online backup with BRMS

Online backup of Domino databases and now incremental online backup on iSeries are implemented with the IBM licensed product Backup Recovery and Media Services for AS/400 (BRMS), available for OS/400 V4R4 and V4R5 as 5769-BR1 and for V5R1 as 5722-BR1. Online backup implies that QuickPlace, Domino, and other Lotus server databases on the iSeries can be saved while they are in use, with no save-while-active synchronization points. This is true online backup support.

Note, that incremental online backup needs archival logging and cannot be used if you leave the default of circular logging style.

Circular or archival logging style

Circular transaction logging is designed for fast and efficient recovery after a crash, to improve the performance (not necessarily for iSeries, see earlier explanation), and automatically re-use the log files in rotation. Only the *archival logging* style can be used for backup purposes, because the log files are not overwritten before they are saved. See “Setting up transaction logging” on page 379 for information on setting up transaction logging and selecting archival logging style.

Full backup

With a new Domino API, BRMS (or other backup suites) can instruct Domino to flush a database to disk immediately, overriding the “performance” buffering options (see “Configuration parameters related to transaction logging” on page 379). This is essential because the backup suite will copy the Notes database physically from disk via the operating system command set, and will *not* use the Notes interface that would always read all the data from the UBM virtual image in memory. The operating system will read the possibly unflushed disk based copy, therefore, the need to force a flush.

Incremental backup

BRMS can also perform “incremental backups” by querying Domino about which logs are “unprotected” (not backed up) but “available” (that is not the current log) using API commands like NSFGetFirstLogToArchive(). This allows the software to back up the TxL log files from the oldest to the most recently created in date order, therefore “protecting” all log files written to since the last incremental backup. Once they have done this, the backup suite must notify Domino which of the Archival logs has just been successfully backed up (“protected”) and allow Domino to rename and re-use them when it is ready to do so. The one log file that will *not* be offered for backup by Domino is the presently open “active” log file.

Section 6.8, “Online backup with BRMS” on page 178, contains some considerations regarding the performance during BRMS online backup. For more information about Domino online backup with BRMS, refer to the online incremental backup support site at: <http://ibm.com/servers/eserver/series/service/brms/domIncremental.htm>

13.5 Miscellaneous tips for Domino transaction logging

The following sections contain several, not necessarily performance related, tips on Domino transaction logging.

13.5.1 What is the Database Instance ID (DBIID)

When you enable transaction logging, Domino assigns a *Database Instance ID* to each Domino Release 5 database, that is all databases with ODS 41. When Domino records a transaction in the log, it includes the DBIID. During recovery, Domino uses the DBIID to match transactions to databases (it identifies to which database the changes should be applied). The DBIID is stored in the file header, along with the database ID and the Replica ID. Note, there is no relation to the Replica ID or the database ID (DBID).

Some database maintenance activities, such as compaction with options, cause Domino to assign a new DBIID to a database. From that point forward, all new transactions recorded in the log use the new DBIID. However, any old transactions still have the old DBIID and no longer match the database's new DBIID. As a result, Domino cannot restore these old transactions to the database.

To avoid losing data, you should immediately perform a full database backup whenever a database receives a new DBIID. When you perform this backup, you capture all the database transactions up until that point and ensure that Domino needs only the new transactions (with the new DBIID) to restore the database. If the DBIID changes and a backup is not taken after the fact, the database cannot be successfully restored (backup will have the old DBIID and the transactional log will not “know” the old DBIID).

Changes to DBIID

The Database Instance ID is generated when transaction logging is enabled. Domino assigns a new DBIID to Domino Release 5 databases when:

- ▶ You enable transaction logging for the first time.
- ▶ System logging is disabled and then re-enabled.
- ▶ Fixup is forced on the database (fixup -j).
- ▶ You move an ODS 41 database from one logged server to another logged server or from an unlogged server to a logged server.
- ▶ The database is compacted copy-style.

The DBIID changes when a database is copy-style compacted because it essentially creates an entire new NSF with a new structure, which basically does not match the structure in the logs for the “old” NSF anymore. You should use the new inline COMPACT method of COMPACT -b (with lowercase -b).

Compacting a Domino database

A Domino database can be compacted using the Domino console command **load compact databasename -switch**, where *switch* defines the type of compaction to be performed. Since transaction logs cannot be applied after the DBIID was changed, it is important to understand when COMPACT changes it. The following rules apply to DBIID by the COMPACT task:

- ▶ If a database is logged, the default for COMPACT with no switches is -b (lowercase)
- ▶ If a database is unlogged, the default for COMPACT with no switches is -B (uppercase).
- ▶ COMPACT with no switches and COMPACT -b (lowercase b) are the only times COMPACT does not change the DBIID.
- ▶ The DBIID changes when a database is copy-style compacted because a copy-style essentially creates an entire new NSF with a new structure, which basically does not match the structure in the logs for the “old” NSF anymore.

Note: -L, -c, and -i are switches that enable copy style compaction. -B at times uses copy style compaction.

- ▶ COMPACT -B may change the DBIID. This option uses in-place compaction unless there is a pending structural change in which case copy-style compacting occurs. When you use this option and transaction logging, do full database backups after compacting completes.

For additional information on the COMPACT tasks switches, refer to the Technote “Switches for COMPACT Server Task for Domino R4 and R5 Document” (#172454).

13.5.2 How transaction logging technically works

Although it is new to Domino, transaction logging is a mature technology that represents the state of the art in reliable data storage. It is based on *Algorithm for Recovery and Isolation Exploiting Semantics (ARIES)*, developed by Almaden Computer Science at IBM, and is implemented in Domino by a joint team of developers and researchers from IBM and Iris Associates.

Lotus Domino R5 is the first semi-structured data store that implements transaction logging. The ARIES algorithm has been in production since 1988. Versions are used not only in Domino R5 but also in IBM DB2 mainframe and workstation products and IBM MQSeries, among numerous other products. (See the Lotus white paper “Why you need transaction logging” at <http://www.lotus.com>)

So what's that? Simply put, all updates to a Domino database are actually a series of “transactions” between the user and the Domino database engine, using our Domino-defined procedure calls that we must use to make any changes to the data within the database. There is no other way to update the Domino database. That is how our security and database integrity are managed and are the core of our product.

We tried to log the actual data itself at the field or record level to some log file, and then tried to determine how to roll these changes back later, perhaps by comparing the data byte-by-byte with the database being recovered. This was painfully slow. Instead, with Domino transactional logging, we logged both the data and actual transactional commands used to make the change, for example “NSFNoteOpen, NoteUpdate, NoteClose”.

Therefore, we log the actual commands used, *plus* the data itself with date and timestamps, etc., to create a complete series of “replayable transactions”. We don't really log words like “NSFNoteOpen”; but we log in binary, of course, and use our command offsets to minimize log file size.

These individual Domino R5 transactions are limited to a maximum size of 16k.

Recovery with UNDO and REDO as a series of data slices

To simplify the picture you are building here, we wrap our series of 16k Domino transaction data slices between two matching “crusts”, the UNDO and REDO records. However, we don't actually write them to the log file until we are ready to commit the whole series of transaction records.

If during recovery we find an UNDO record, then some transaction slices, then a matching REDO record, we then make certain that the hardened database contains all the transaction data slice records.

If during recovery from backup we also lose the most recent log files, we may find an UNDO record but no matching REDO record. In this case, we reverse the process with all the remaining transaction data slice records and discard all the data slices.

If the data sliced transaction records following the UNDO record (with no matching REDO record) were never in fact hardened into the database, then during the recovery we can simply ignore them.

Maximum log file sizes

Currently Domino can only create a maximum of 4 Gb of circular log files. With archival logging, Domino can currently only pre-allocate 4 Gb. So if you do not do any backups, it creates new log files as it goes along after it reaches 4 Gb if none of the log files (current contents) have been “protected” by being flagged as backed up. Archival logging continues to do this until the disk space on the notes.ini variable Translog_Path= runs out. Archival logging ignores the Translog_MaxSize in this situation. (As you may recall, it used it for the pre-allocate.)

Be aware that there is a recommended (supported) limit of 4 Gb of uncommitted log files. This equates to a supported maximum of 70 extent files (4.48 Gb) for your backup software to protect and then release back to Domino.

Protected log file

Therefore, as the next log is opened for use, the just completed log is flagged as “protected” and available for backup. The backup API can also locate the current (open) log file using NSFGetFirstLogToArchive() and NSFGetNextLogToArchive() to return LogPath of any transaction log whether it was waiting to be archived. Backup software vendors have the ability to back up the current (active) log file as well.

Archival logs and the 4 Gb and 70-log limit

Archival transaction logging is used for partial and full online Domino database backups. Archival can pre-allocate 4 Gb (same as circular logging) but archival can then continue logging additional extents way beyond 4 Gb until it runs out of space if you don't run a backup. We currently only recommend and support a maximum of 4 Gb of unprotected (unbacked up) data in the TxL logs.

The Domino TxL API currently passes the names of only the first 70 unprotected log files. The backup mechanism needs to know to ask again after the first 70 for the next 70 or remaining files and so on. This may result in leaving an unknown number of TxL logs (if there were more than 70) unprotected. Also note that a bug that was reported as fixed in R5.0.9 actually prevented successful backup of more than 70 unprotected logs even if the software knew to ask twice! See SPR SCON4W3UZD on the Web at:

<http://www.lotus.com/home.nsf/welcome/support>

While Domino may not crash or stop, be very careful if you step over the 70 logs or 4 Gb TxL unprotected log limit, especially before Domino R5.0.9. Your backup software may not do what you expect, and you risk not backing up all the unprotected TxL files. They are not overwritten because the backup software doesn't "release" them. They just aren't backed up as part of your backup strategy and lie around forever! This limit will be increased in future revisions.

The 64 MB exposure between last closed and the open TxL log file

Every transaction hardened to the Domino databases disk and caught in the backed up log files is not necessarily complete and may not be recoverable in the event of a crash and restart, unless you look at the boundary.

Suppose we have an UNDO in the recently closed TxL log, with the matching REDO in the current (open and active) TxL log. Any data in the open "live" TxL log is not backed up (unless we also take the current open (partial) log file that we can find with the API). The log containing the final REDO will have to wait for the *next* backup cycle before it is fully secured to the backup media. Therefore, if we now lose all the log files, we will try to roll back any transaction spanning the "backup boundary" to remove any partially complete transactions (because we lost the REDO in the crash). This is assuming we don't have a suitable checkpoint to work from. If it's hardened in the database and we have a checkpoint to confirm this, we will not roll it back out. But assuming we have no suitable checkpoint, or we also lost the actual Notes databases, we locate our full database backup and recover from it. As we recover from our original incremental backups, we definitely won't apply those boundary changes because there is absolutely no matching REDO record.

To prevent this (as much as possible), be sure that the backup suite you use must be capable of finding, backing up, and recovering from the partially written "open" or "active" log file ("extent") whenever the backup runs. Or, like the BRMS backup, you must have an add-in task that monitors the active log file and backs it up as soon as it is full, and also backs up what residual data is in the new active TxL log file as well! Just be aware that you are potentially up to a maximum of 64 MB of live data file changes "behind" after every backup if you don't consider this. This is 64 MB of actual changes (not reads or index); it is up to 64 MB of new or changed data, potentially very valuable. Remember, we have to lose the actual TxL log files (disks) in a major disaster for this to be a worry.

Hardening databases and log files

The change from one transaction log to another does *not* relate at all to the frequency of hardening of the database. That is the UBM task depending on that memory image size via the performance factors selected, as mentioned above. And we do not record the hardening in the log file either, not directly.

The occurrence of hardening is actually recorded in the checkpoint snapshot that we write to the log periodically, again depending on the performance factors selected. On a restart/recovery, the UBM will find the most recent hardening time stamp from the actual database on disk (which of course may be our original backup, and not the one it last hardened anyway). From that information, it will search back through the transaction logs for the last checkpoint written to the TxL log for that particular database with the exact matching DBIID.

The UBM then works forward from that matched checkpoint and locates all subsequent transaction log files for that period of time. Then it searches them for transaction records that occurred *after* that last hardening time stamp (on the actual database), and proceeds from there.

The 64 MB log size limit is a compromise between fast sequential writing to a single file versus the number of log files that need to be searched on recovery. Remember, the bigger the file is, the less close and open operations between log files need to happen, and the faster the serial write speed will be. With less records in each log, the sooner we can locate the record with the time stamp we are looking for, the sooner the actual recovery can start.

This “only check for subsequent changes from recent logs” is why it can recover so quickly, because it does not have to “fixup” the entire database. It just has to replay or backout things using the log records created subsequent to the last hardening as recorded in the disk image of the database.

Views and indices are not logged

Views and indices are *not* transactionally logged in R5. There isn't really much point adding overhead and I/O to the log extents when they can easily be re-created after the database is recovered using the UPDALL task.

As a consequence, in Domino R5, after successfully recovering your Notes database from the wreckage or from backup media and replaying the TxL log files, you should run UPDALL -R to recreate all the indices and views again to complete the recovery operation. However, using this option is resource-intensive, so use it as a last resort to solve corruption problems with a specific database. It is rumored that views may be logged in a future release, so watch the release notes.

Other benefits of transaction logging

Transaction logging offers benefits of tracking and capturing all mail for banking, insurance finance and city institutions.

By setting up transaction logging on all user mail files on all mail servers, it is possible for the first time to capture every single piece of mail! TxL records every mail item, plus attachments, as soon as it arrives or is sent out. Even if it is immediately deleted, we can still recover a full copy of all mail items by playing back the transaction log to the point before any mail item was deleted.

With Domino R5's full secure mail tracking, keeping copies of all inbound and outbound mail can now be achieved just by using TxL! Other incremental backup methods can only record the items actually still remaining in the Notes databases at the end of the day and those that were added since the last incremental backup. On the other hand, TxL keeps the full context of all mail conversations, plus attachments, no matter how soon the mail items were erased, even if deletion stubs are purged. This particularly helps auditor in the event of suspected fraud or insider dealing.

Related documents

For more information, consult the following sources:

- ▶ “Why you need transaction logging” white paper from the Technical Library at:
<http://www.lotus.com/home.nsf/welcome/itcentral>
- ▶ Technotes:
 - Transaction Logging in R5 #172508
 - How Do You Check Your Transaction Log Statistics? #173421
 - Commonly Asked Questions About Transactional Logging #177785
 - How to Utilize Transactional Logging Statistics #177796
 - “Circular” Versus “Archive” Transactional Logging in Domino 5.x #179363
 - Transactional Logging and How it Operates #179858



Capacity planning for Domino server using BEST/1

This appendix describes how you can use BEST/1 for capacity planning for Lotus Domino for iSeries. However, BEST/1 will not be supported on OS/400 releases later than Version 5 Release 1 (V5R1). Therefore we decided to move this chapter from the main part of the book to this appendix.

The generalized approach covered in this chapter is simple. However, the approach relies heavily on having good performance monitoring process in place. Essentially, it consists of using historic data on the key aspects of the system and using mathematics and graphing to predict when a particular area of the system's performance exceeds the relevant guideline. However, the approach does not use the sophisticated mathematical modeling techniques that are found in tools such as BEST/1. Consequently, only rough estimates can be determined. These estimates may not apply to all situations. Following the generalized approach, we describe the use of BEST/1 in the context of Lotus Domino for AS/400.

The first section discusses capacity planning concepts in general and offers an in-depth view of how capacity planning is done for a Domino server. General information on workload definitions for Lotus Domino are also provided.

The need for capacity planning

The objective of any capacity planning activity is to estimate the system resources required to deliver performance based on a forecast level of business activity, a specific application, or a specific area of business. The estimate is delivered according to a documented service level agreement (SLA).

Another term that is often used in this area is *capacity sizing* or *application sizing*. This document works on the basis that capacity sizing is the process of estimating the system resource required to deliver performance in accordance with a documented SLA for a forecast level of business activity, a specific application, or a particular area of the business.

Capacity planning prerequisites

You must consider the following factors before you start capacity planning:

- ▶ Work management must be set up properly.
- ▶ A workload profile must be established.
- ▶ The IT environment must be tuned.
- ▶ All components must be below utilization guidelines.
- ▶ History of performance data should be available.
- ▶ Details of new applications should be known.
- ▶ SLA should have been published.
- ▶ Growth data needs to be available.

In the real world, it is rare that all of these prerequisites for a capacity planning exercise are in place. However, if they are not in place, the accuracy of the outcome of the exercise is adversely affected. Therefore, it is important that any shortcomings in these areas are fully documented in the final deliverable from the capacity planning exercise. This allows readers of the plan to make the appropriate allowances.

Basic capacity planning process

The following list highlights the steps for a basic capacity planning process:

1. Build a spreadsheet to summarize collected data.

Review the collected performance data and build a spreadsheet to summarize the data for the peak hour each day. All of the data in the spreadsheet can be obtained from the Performance Tools/400 System Report. However, Domino transactions are not collected as interactive, and therefore, do not show up on the report. The transactions can be found within the Domino REPORTER log file.

The data in the spreadsheet should be organized to allow easy assimilation of the key aspects of workload and performance. The following list identifies information to put in the spreadsheet:

- **System level:** An overall view of performance given by total CPU utilization and interactive response time. However, interactive response times are only useful for non-client/server interactive workloads. This is true for Domino jobs as well.
- **Applied workload:** The workload that is customized to the particular environment. For Lotus Domino, a workload can be defined for each functional area of work such as Calendaring & Scheduling, mail use, Web serving, and each specific Domino application. The Applied Workload section of the spreadsheet is expanded for each area of work. The information within this section is found in the System Report when

printed, if the workloads are separated by subsystem. If this is not the case, the workload areas must be calculated based on the different types of jobs running. This is a manual process, which increases the amount of possible errors.

- **CPU breakdown:** The first of three utilization figures that should be monitored carefully. The high priority CPU figure is usually the figure that is watched closely because if it exceeds its threshold figure, high priority work is affected.
- **Main storage:** The guideline figures here obviously depend on the model, but they provide a quick check. If the figures approach the guidelines, then further investigation should be instigated.
- **Disk performance:** Both arm utilization and space utilization need to be monitored since they can adversely affect performance if they exceed their guideline figures (50% and 85% respectively).

We are only considering capacity planning for the iSeries server, and are not taking into account any network issues. Therefore, we have ignored any communications-related usage figures.

2. Calculate weekly averages, and plot “best fit” lines.

For each week, determine the average set of values for the week. Review them to see which of the three crucial values (CPU, main storage, or disk drives) is closest to its threshold value. Extract this number into a separate worksheet.

From the worksheet, calculate the coordinates of a simple best-fit line for transactions per hour and for high priority CPU percentage. You may also want to graph transactions per hour and high priority CPU utilization.

3. Forecast when an upgrade is required.

Zero in on the high priority CPU. An increase in this over a period of time dictates when you need to upgrade. As long as the increase is linear, and workloads are not added or removed, the time frame in which to upgrade can be reasonably predicted. Be careful not to track the wrong data. A trend at month end does not provide an accurate view of your overall CPU utilization. It is important to focus on the peak periods of high priority CPU utilization.

These three simple steps show simple capacity planning. They do not answer the questions that any business usually asks with regard to upgrades to their system. These questions are:

- ▶ What upgrade is required and what will it cost?
- ▶ How long will the upgrade last?

To answer these questions, we leave behind our simple graphs and best-fit lines, and use the BEST/1 modeling tool.

CPW values versus Domino mail user ratings

The Commercial Processing Workload (CPW) is a synthetic workload used to measure the relative throughput capabilities of the iSeries family. This workload may not be representative of any particular customer environment. The overall throughput in any processing environment results from the combined affect of hardware capability and operating system efficiency.

Although the CPW workload modeling still holds true for traditional interactive (5250) applications with high I/O and relatively low processor demand, it may not be the best modeling tool available for sizing Domino workloads. New sizing ratings like Compute Intensive Workload (CIW), Simple Mail Users (SMU), Mail and Calendaring Users (MCU) and Typical users may be more beneficial in sizing machines for Domino workloads.

CIW ratings are modeled projections based on the internal workloads of such application as Domino, SAP, and J. D. Edwards where most of the workload is spent inside the application performing instructions versus disk I/O. CIW workloads tend to spend a lot of time computing and less time doing I/O. You can find more information on CIW in the *iSeries Performance Capabilities Reference Version 5 Release 1*, SC41-0607.

SMU and MCU workload ratings were calculated in an automated environment similar to what is done in the mail workload from Lotus NotesBench.

SMU was driven by a NotesBench mail workload on Domino 4.6. This workload was based on the user performing the following tasks every 15 minutes on average:

- ▶ Open mail database (documents <1K)
- ▶ Open view
- ▶ Open five documents in the mail file
- ▶ Categorize two of the documents
- ▶ Compose two mail memos 1K in size every 90 minutes
- ▶ Mark documents for deletion
- ▶ Delete documents
- ▶ Close views

The *MCU* workload came along with the transition to Domino R5. This workload is heavier than SMU due to the fact that it contains calendaring and scheduling functions. The MCU workload is similar to SMU in respect to what the user will do on average every 15 minutes, but at a greater intensity. The following tasks are performed:

- ▶ Open mail database (documents <10K)
- ▶ Open view
- ▶ Open five documents in the mail file
- ▶ Categorize two of the documents
- ▶ Compose two mail memos 10K in size every 90 minutes
- ▶ Mark several documents for deletion
- ▶ Delete documents
- ▶ Create one appointment every 90 minutes
- ▶ Schedule one meeting every 90 minutes
- ▶ Close views

Both SMU and MCU workloads are not heavy enough workloads to depict real life users. Therefore, the *typical mail workload* was defined. It is used in the Workload Estimator (WLE) and several publications comparing iSeries performance capabilities. This workload is three times heavier than SMU and two times heavier than MCU. Based on the above information, if you have one rating, you can calculate the other two ratings. Here are a couple of scenarios:

- ▶ If a given server supports *X* typical mail users (and we rate these at 70%), the same system will support 3x SMU and 2x MCU.
- ▶ A 270-2423 has a rating of 2570 MCU. This server would support 1285 typical mail users ($2570/2=1285$) and 3855 SMU users ($1285*3=3855$).

You can find further information on these ratings in Appendix A of *iSeries Performance Capabilities Reference Version 5 Release 1*, SC41-0607. You can also find CPW and CIW ratings for iSeries servers in Appendix D of the same manual.

Attention: Although this is called a *typical* mail workload, the *real* workload may vary extremely from one customer environment to another and even between users within the same environment.

Capacity planning unit of measure

All work performed by the AS/400 processors can be considered as tasks. Each task represents one or more requests for service and associated responses.

Each request and response is an identifiable unit of activity. In this chapter, we refer to this as a *capacity planning unit of measure* (CPUM). The boundaries of a CPUM can be determined according to the needs and measurability afforded by the specific Domino application being considered. You are free to select the most suitable set of boundaries for the request and response, based on measurement facilities available in the business environment, the application suite, and the AS/400 Domino server.

For example, a traditional batch job can have a request such as the complete processing of specific input data sets or the achievement of specific completion criteria. In this case, the boundary of this CPUM is the beginning and end of the batch task with a CPUM value of 1. The response time is the elapsed time of the task.

In the case of an interaction with a Notes user or client device, the request is initiated by pressing Enter, a function key, or making a selection with the mouse. The response to this request completes the CPUM. This is similar to a transaction referred to in interactive iSeries performance measurements. The computer systems, communications activity, and communications line turnarounds that occur within a CPUM depend on the specific client/server or interactive application.

In terms of determining Domino transactions, the most appropriate or easiest way to do this is to collect the STATS information in the REPORTER log file. You can also run the **show stat** command for a running transaction total. Doing this periodically (every 15 minutes) should give you a transaction and throughput value that you can use as the CPUM. This transaction value is at a more functional level and is not related to transactions made by pressing the Enter key. Still, this information can be used for modeling, because, as the business transactions increase, the transactions at a functional level should also increase.

Business transaction

It is important to distinguish the request and response interactions between the Notes user and the client, or between the client and the Domino server from a business transaction. A business transaction is made up of multiple user interactions to complete a single business transaction such as an e-mail transmission, calendaring, or Domino application. One of the fundamental points to cover in capacity planning, is the relationship between a business transaction and a system transaction. When this is identified, the growth analysis is done in a proportionate fashion.

Measurement

In certain environments, the application software may provide precise measurements of numbers and response times of each window change at the client. The measurement of response time is taken at the Notes client or at the Domino server.

The primary focus when dealing with measurements is that there is an assumption made that the system and Domino server are well tuned before any capacity planning is performed. In particular, the utilization of the Domino server must be within the recommended guidelines of the system. If not, a non-linear result may occur and the results of the capacity planning may be unpredictable.

Domino server workload definition

One of the essential points in capacity planning is to determine the different types of work that are being performed on the system. Looking at the transactions being processed on the system, you can categorize each transaction into specific groupings. These groups contain *like* transactions relative to what the system is processing. Each group has defined a basic workload and describes similar characteristics. The following section describes in general what these different workloads are for Domino.

What the different workloads are

As mentioned earlier, a workload is just that, a specific definition of work being loaded on the system. All the jobs or tasks on the system are categorized into certain workloads. This is useful for capacity planning because these workloads usually relate to a business function. The workloads can also be modeled using business terms that a customer can easily understand.

For example, you can group Domino into four key workloads: Domino Mail Users, Domino Calendar & Scheduling Users, Domino Web serving, and Domino Applications. A fifth workload that you cannot ignore is any non-Domino application running on the system. For the scope of this appendix, we assume that non-Domino applications are tuned and modeled appropriately.

Having these areas of work defined helps explain where the resources of the system are used. However, there is a difficult problem that shows up when attempting to define workloads in a client/server application. It is not possible to directly map the relationship between the client performing the work and the resources being used on the server. With the open ended architecture to which all platforms are moving, there is a problem in tracking client to server transactions. Therefore, modeling transactions per user, or even per workload can be difficult if not impossible.

As described in this chapter, one of the best solutions to this problem is in setting up workloads relative to subsystems. This process is easily done with Domino, because each Domino server is a subsystem. Therefore, you simply set up a server for each type of workload. For example, Server1 is mail users, Server2 is Domino Web serving, and Server3 is Domino application. Each server has its own subsystem, and therefore, allows you to monitor the different workloads at a functional level.

Note that in a client/server environment, we do not typically associate a single job running on a server for the client. Instead, a single job running on the server may be used by multiple clients. This is because the client is not technically running on the server, and therefore, does not have a unique job solely owned by it. In a traditional green-screen environment, the client resided on the server, and therefore, owned a portion of the resources. In today's client/server environment, the client only borrows enough resources to get its work done and attempts to process what it can by itself. Therefore, Notes clients come in and perform work in generic job names, making it virtually impossible to gather performance information at the job name level in the traditional way.

Domino mail user

Attempting to define in detail what a Domino mail user workload includes, in actuality, is an overwhelming task that is not useful in a real environment. This is due to the complexity of the mail user itself. You may have a simple mail user that only sends quick messages to a few users in their department. Or, you may have a user that sends long messages to multiple users with attached documents. In another situation, you may have a mail user that sends hundreds of e-mail messages a day.

There is a complexity to the definition. In general, only a few functions show what a mail user is doing. The user deals with database opens and closes, as well as routing if the mail is being sent through the Internet or intranet. Although this gives you information on what mail users are doing from a functional level, defining a relationship between functions and transactions for capacity planning may be a difficult task. Therefore, we recommend that you associate mail users with a server if you want to monitor performance information for the mail user workload.

Domino Calendar & Scheduling

Domino Calendar & Scheduling brings about another type of workload for Domino. This workload depends primarily on the frequency of use. Furthermore, the general area of work being performed is with text searches when attempting to schedule a meeting for users. As the number of scheduled meetings increases, the work for this workload increases.

Although text searching is the key point in processing for this workload, determining a relationship between Domino Calendar & Scheduling transactions and text searching may be a difficult task. Therefore, we recommend that you place the Calendar & Scheduling workload in its own server. This allows you to track transactional information by server, which, in turn, shows you transactions by workload. This can be used for proper capacity planning.

Domino Web serving

Determining the Domino Web serving workload is one of the more complex tasks. The resources used by the system heavily depend on how your Web site is setup. If you have a simple home page setup with static pages, a more straightforward capacity plan can be used. However, this changes when dynamic pages are involved, database lookups are used, and complex intelligent agent processing is performed. This change affects the complexity of the workload and makes it more difficult to plan for capacity. As with the other Domino workloads, we recommend that you set up Web serving on its own server if gathering workload performance information is necessary.

Domino user applications

Domino user applications can seem like a vague term at first. In general, all Domino user applications center around one concept: document processing. Although there are many different ways to access, create, delete, and maintain Domino documents, they all relate to database manipulation.

Notice that some applications written for Domino may not center around this concept. In this case, take extreme care in reviewing the application, especially if the application is trying to focus on an online transaction process environment. Although Domino can easily handle online transactional processing, be aware of the additional overhead in CPU processing. Other than this, a Domino user application generally requires heavy database open and close functions. As with the other workloads, the best way to monitor performance information for capacity planning is to have the Domino user application in its own server.

Splitting up the workloads

As indicated in the previous section, break the work being performed in Domino into different workloads. Each workload can be placed in its own server. In doing this, there is a relationship between the workload and subsystem for each Domino server. This allows you to gather performance information relative to the subsystem. You can use the iSeries performance tools reports to review information by subsystem.

In many cases, setting up multiple subsystems may not be possible because of the size of the system or the requirements of the business. In such a case, it is increasingly difficult to capture pertinent information relative to performance by workload. This is not to say that it is absolutely necessary. However, it is more convenient for capacity planning purposes. In the worst case, capturing performance data for the single Domino server still gives you accurate information, but not at the workload level. You can still use the data for capacity planning, but it may not be as accurate.

Other Domino workload concerns

The workloads mentioned in the previous section describe, in general, what Domino is made up of in terms of the work being performed. However, there are other areas of work that Domino performs that are not covered. These functions tend to run in the background and are not associated with a particular transaction. They are database replication, clustering, using multiple databases, and non-Domino applications. The following sections briefly describe how the different functions affect performance and, in turn, capacity planning.

Database replication

Database replication primarily affects performance and capacity planning through its database open and close functions. As a database is replicated, the database is opened and closed to copy the data. This process in itself is not too difficult to monitor. However, it is important to know when replication is occurring so you can ensure that proper performance data collection is taking place. Replicating a database may skew the collected performance data. If you are not aware of this, inaccurate capacity planning may result.

Clustering

Clustering is another area of work that can affect the results of performance collection for capacity planning. With clustering, additional overhead is involved to handle the failover processing that can occur. Furthermore, you must be aware of clustering on the system and ensure proper data collection if failover has occurred during performance collection. The data collected during a failover does not represent typical performance for that server.

Using multiple databases

Using multiple databases may be more of a necessity than an option. In some cases, you may need to split up your data among multiple databases, depending on the workload. Dividing your data over several databases may improve performance by reducing the time Domino takes to index over a single database. Multiple requests for a database may have to wait for a previous request to complete if they are all accessing a database for updating. If multiple databases were used, multiple requests may gain access faster. We recommend that you keep a more simple view by using only one database, unless definite performance improvements are possible.

Non-Domino applications

The key point in observation here is to know your entire system workload. Make sure you understand if there are applications other than Domino running on the system. These other applications can adversely affect Domino performance and capacity planning, depending on what resources they require. Also, note whether you will add non-Domino application workloads in the future.

BEST/1 for Domino server capacity planning

The objective of any capacity planning effort is to determine the optimum, cost-effective configuration required to support a specified workload (typically based on currently measurable levels of activity). This configuration must ensure that acceptable response times are delivered to the user.

This section presents an overview of the factors that you need to consider before launching a capacity planning project. These considerations are valid for most capacity planning efforts and are not confined to an iSeries server environment. With this in mind, the same capacity planning process can be used for a Domino Server with the exception of a few points that are described in this section.

While it is not within the scope of this section to discuss performance tuning and performance optimization, we strongly recommend that you make every effort to ensure that the Domino server and other applications on the system are as efficient as possible. If this is not done as a prerequisite, any problems associated with the inefficiencies are carried through with the increased workload, causing increased resource overhead.

Attention: After V5R1, BEST/1 will no longer be a part of Performance Tools and will no longer be an offered Licensed Program Product for OS/400.

Introduction

Capacity planning is a process to determine future computing hardware resources to support the estimated changes in computing workload. The increased workload on computing resources can be a result of growth in business volumes or the introduction of new applications and functions to enhance the current suite of applications.

The results of capacity planning are only an approximation at best. The implementation of the same application in different environments can result in significantly different computing system requirements. There are many factors that can influence the degree of success in achieving the predicted results. These include changes in application design, the way users interact with the application, and the number of users who may use the applications. It is also difficult to determine external factors that affect the distribution of the workload over a given period of time, such as phone-in customer orders during a working day.

The objective of this section is to present techniques to assist in the prediction of iSeries system requirements where the iSeries server functions as a Domino server in a Notes client or Domino server computing environment. No attempt is made to determine client or network requirements. The functions of the iSeries capacity planning facility (BEST/1) are available as part of the iSeries Performance Tools Licensed Program (5722-PT1) Manager feature. The functions are used to predict iSeries requirements. Detailed information about the use of these functions is available in *BEST/1 Capacity Planning Tool Version 5 Release 1*, SC41-5341.

Note: There were no functional enhancements to BEST/1 in V5R1. However, revisions were made to the documentation.

This section introduces the key aspects of BEST/1 that need to be considered in capacity planning projects for iSeries Domino server tasks and applications. The objective is to help you become familiar with the relevant BEST/1 facilities.

The suggested approach assumes that performance data collected with the Collection Services is available to build the model. However, the Collection Services does not record non-5250 transactions. Therefore, there is no granularity in the measurement of resource usage in an iSeries Domino server environment. The iSeries performance measurements do not recognize individual requests to or responses from a job running on a Domino server. Also, all Domino server tasks are “batch-like” and no LAN or WAN times are attributed to them. Therefore, it is not possible to use the iSeries measured performance data for capacity planning if you require some response time indications.

Any references to transactions and response times from reports produced by Performance Tools for iSeries (5722-PT1) refer to interactive work produced by 5250 sessions. They do not apply to any Domino server applications that may be running at the time. The reports produced using the standard Performance Tools options provide you with information on the total use of resources, such as processor, disk, and so on for server jobs. However, you may use the Change Job Type (CHGJOBTYP) command to change the job type for batch jobs to *interactive*, and run the performance reports to make the appearance of interactive workload. This gives you interactive information. However, it can be misleading and a skilled performance specialist should be involved to interpret the results.

iSeries Domino server modeling

There is no intent to discuss designing Domino applications for performance, performance management techniques, or system tuning options. The projections start from a known, measurable workload and extrapolate the resource utilization and response times using a specified rate of growth.

The basic prerequisites are:

- ▶ The system and all applications run optimally at expected levels of performance both for Domino and non-Domino applications.
- ▶ There are no application-dependent constraints that invalidate BEST/1 extrapolations.
- ▶ No changes are expected in patterns of application usage or user behavior.
- ▶ A particular period of measurable activity on the iSeries server is established as being representative of a sustained high peak workload within recommended guidelines.
- ▶ The proposed increase in workload is related to the activity measured.
- ▶ An acceptable average response time estimate is established.
- ▶ A suitable capacity planning unit of measure is determined based on application knowledge and measurability.

The key topics covered in this section include:

- ▶ Model creation using measured data for a non-interactive workload
- ▶ Model validation
- ▶ Growth analysis

Assumptions

The capacity planning process makes the following assumptions:

- ▶ The Domino server and applications under consideration are optimized for the environment it is running in with regard to:
 - Functions
 - Business requirements
 - User behavior
- ▶ The performance measurements used in the capacity planning project are a good representation of a typical busy workload on the system, including the mix of activity and volume of work.
- ▶ There are no Domino servers or application-dependent bottlenecks that prevent growth in throughput or improved response times. For example, an application may support only a single communications I/O processor, or there may be a common code-path within the application that only allows requests to the application server to be single threaded.
- ▶ The performance data is collected on a system that has not reached a saturation point with regard to any key system resources such as memory, processor, disk, and so on. When system resources are saturated, they introduce overhead, such as queuing, that is difficult to isolate and eliminate in the modeling process.
- ▶ The nature of the Domino server or applications and its complexity do not change significantly as a part of the capacity projection. For example, a version upgrade of an application suite often results in significant changes in resource requirements associated with additional functions, processing architecture, and user behavior.
- ▶ There is no change in the overall use of any functions of the Domino server or applications that results in increased utilization of system resources.

If any of these assumptions is not valid, plan to make allowances for these increases in the projected workload increases.

Understanding the Domino server environment

Consider the multiple areas of work within a Domino server environment. Several different workloads are collected together using the same underlying resources. An increase in one workload may cause a cascading effect on related server jobs as well as an increase in their workload. For example, a planned increase in mail users, due to an acquisition of a company, increases the workload of the Domino server tasks and may increase workload on a non-Domino sales application.

You also need a good understanding of the overall architecture of the system and its applications, both Domino and non-Domino. For example, you need to know if the Domino application uses a distributed logic approach with a significant amount of processing performed at the client. Associated with this type of application, there may be significant workload on the communications bandwidth resulting from high data volumes and line turnarounds. This knowledge enables you to determine the impact of the capacity of the iSeries Domino server on the total end-to-end client/server throughput and response. It also assists you in setting levels of expectation with regard to results that may be achieved once your recommendations are implemented. A good understanding of the Domino server and applications can also assist in taking measurements that enable you to create a reasonably accurate workload model for extrapolation.

Selecting a capacity planning unit of measure

Identifying a measurable capacity planning unit of measure is critical to the success of a capacity planning project. The selected unit of measure must be clearly understood and have boundaries that can be defined and identified.

Measurability

It is important to measure a specific workload with regard to:

- ▶ Response time per CPUM
- ▶ Number of CPUMs per unit of time
- ▶ Components included in the measure
- ▶ Overall iSeries resource utilization such as:
 - CPU
 - Memory
 - Disk

The capacity to measure within a Domino server can be difficult to capture because of the broad functional areas that the individual tasks perform. A server task can process for a mail user at one moment and an application user the next. Attempting to define workloads is made easier if the workloads are partitioned across multiple servers to ensure that each Domino server supports a separate workload. In doing so, capacity measurements can be found within the performance data collected and viewed by the subsystem. It may not be possible to completely separate the workloads into different subsystems. Even if most of the workloads are separated, you can more accurately determine capacity planning.

One other aspect in collecting capacity planning measurements is the inability to easily collect response time information. This, like most client/server environments, presents a problem when it comes to capacity planning.

In some cases, you may want to use business transactions, such as Domino e-mail read or send, document lookup, or Web browsing, as a capacity planning unit of measure. The volume of transactions is easily calculated for a period of time. However, each business transaction is the product of multiple interactions with the computer system with some amount of key think time between each interaction. In this situation, the capacity planning model must make allowances for these external delays.

Determining growth objectives

An effective capacity planning project requires an estimation of the proposed increase in workload with which the computing system must cope. Often the business managers predict increases in terms of sales revenue, profit projections, or other business measurements. These business measurements must be translated into the impact on the computing workload.

You must work closely with the customer to establish these increases and gain agreement on the estimated increases. A simple increase in volume at one level of the business can have a cascading effect on the overall computing activity on the system. Ensure that you document the assumptions and the basis on which the increases in computing workload were determined. This helps in reconciling any differences that may be encountered when the changes actually occur.

Setting expectations

A clear understanding of the expected deliverable of the capacity planning project must be agreed upon. It is important to stress the predictive nature of the process, and the fact that there are many factors that can influence the degree of success in achieving the predicted results.

There are margins for error at every stage of the predictive process including:

- ▶ Estimation of workload at the business level
- ▶ Existence of application dependent bottlenecks
- ▶ Translation of the business workload into CPUMs

- ▶ Prediction of user behavior
- ▶ Determination of periods of peak activity

For these reasons, the conclusions and recommendation of a capacity planning project can only be an approximation. The customer should also indicate expected average response times. It is normally the practice to specify response time requirements below a particular value (for example, under 1.5 seconds) for a specified percentage of the transactions (for example, 90% of the interactive transactions).

Averages: The BEST/1 modeling facilities use averages. Their predictions of utilization and response times are also averages.

Measuring workload and system resource utilization

This is a critical aspect of the capacity planning project because it forms the basis of extrapolation for the growth estimates provided by the customer. Note these points:

- ▶ The duration over which performance data is collected should provide representative information over a period of high activity. If the customer cannot identify a period of sustained peak activity, you may have to measure performance data over a period of time to establish a suitable time frame from which to develop a capacity planning model.
- ▶ Select a capacity planning unit of measure that can assist you in modeling the growth. The selected CPUM must be measurable in terms of the quantity and the response time over the measured period.
- ▶ Identify the various components of response that contribute to the CPUM response time so that the necessary adjustments can be made when determining the iSeries resource contributions.

Building a workload model for BEST/1 requires the following basic information:

- ▶ iSeries resource utilization information
- ▶ Domino server workload measurements:
 - Number of capacity planning units of measure (CPUMs)
 - Average response time per CPUM
 - iSeries resource category used by the Domino server

The first step is to collect iSeries performance data and Domino server workload statistics for the selected period of system activity. In a Domino server environment, the information is collected from more than one source. Most of the required data comes from the performance data. Additional information, specific to workloads, comes from the log files within Domino. See Chapter 5, “Domino logs and statistics” on page 127, for more details.

Collecting iSeries performance data

To collect iSeries performance data, see Chapter 4, “Basic concepts of performance analysis” on page 53, for details.

Collecting Domino server workload data

The major components of the Domino server environment that you need to capture are:

- ▶ Number of capacity planning units of measure (CPUMs)
- ▶ Average response time per CPUM
- ▶ iSeries resource allocation category used by the Domino server (such as job numbers, subsystem name, and so on)

Note: Obtaining this information requires an understanding of the Domino server environment.

Creating a model using measured data

BEST/1 provides the facility to use measured iSeries performance data in building a model for capacity planning. The objective is to create separate workloads within the model to represent the various Domino server tasks and applications that are involved in the capacity planning project.

After the separate workloads are created, you can review them for the effect of independent rates of growth on resource utilization and response time. After the workloads are created, they can be saved and used in other BEST/1 models combined with other workloads.

The workloads can be defined using the following categories or groups:

- ▶ User ID
- ▶ Job type
- ▶ Job name
- ▶ Account code
- ▶ Job number
- ▶ Subsystem
- ▶ Memory pool

For Domino workloads, we recommend that you categorize data by subsystem if your servers are split up by workload.

The major steps in using BEST/1 to create a model with one or more distinct workloads based on your definitions are:

1. Use the Start BEST/1 (STRBEST) command to activate the advanced user level of BEST/1.
2. Select the performance data set to be used in modeling.
3. Identify the time period to be considered.
4. Classify the Domino server jobs into suitable workloads.
5. Define the number of CPUMs per hour for each Domino server workload.
6. Create the model.

The remainder of this section shows the steps in building a BEST/1 model and validating (calibrating) a model.

Start BEST/1

Sign on the system and start BEST/1 by typing the STRBEST command. Press F4 to receive a prompt for input.

Start BEST/1 (STRBEST)

Type choices, press Enter.

BEST/1 data library	a	Name, *CURLIB
Performance data library	b	Name
Log member	*NONE	Name, *NONE
Log library	*BESTDTAL	Name, *BESTDTAL
User level	*ADVANCED	*ADVANCED, *BASIC

Bottom

F3=Exit F4=Prompt F5=Refresh F12=Cancel F13=How to use this display
F24=More keys

Figure A-1 Start BEST/1

In the following parameters (shown in Figure A-1), enter the information as indicated:

- **BEST/1 data library:** Enter the name of an existing library to contain BEST/1 objects created during modeling.
- **Performance data library:** Enter the name of the library that has the previously collected performance data.

An initial menu appears after the disclaimer display is shown.

BEST/1 for the AS/400 system

On the BEST/1 for the AS/400 menu (Figure A-2), select option 1 (Work with BEST/1 models).

BEST/1 for the AS/400

Select one of the following:

1. Work with BEST/1 models

5. Create BEST/1 model from performance data

10. Work with results

50. About BEST/1

60. More BEST/1 options

Selection or command

==> 1

F3=Exit F4=Prompt F9=Retrieve F12=Cancel

Figure A-2 BEST/1 main menu

Work with BEST/1 Models

Select option 1 (Work with BEST/1 models) and you see the display shown in Figure A-3.

Work with BEST/1 Models				
Library	CHRIS	Name		
Type options, press Enter.				
1=Create	3=Copy	4=Delete	5=Work with	6=Print 7=Rename
Opt	Model	Text	Date	Time
1	YOURMODEL			
				Bottom
Command				
===>				
F3=Exit	F4=Prompt	F5=Refresh	F9=Retrieve	F12=Cancel
F15=Sort by model	F16=Sort by text	F19=Sort by date and time		

Figure A-3 Work with BEST/1 Models

Select option 1 to create a model. Enter a name for the model you want to build.

Create BEST/1 Model

The display in Figure A-4 appears. Select option 1 (Create from performance data). You can specify option 2 and use the predefined workload for Domino under the SERVER category. This definition is used as a best estimate. It centers around planning for simple mail users but does not allow for any further complexity.

Create BEST/1 Model	
Select one of the following:	
1. Create from performance data	
2. Create from predefined and user-defined workloads	
Selection	
1	
F3=Exit	F12=Cancel

Figure A-4 Create BEST/1 Model

Create BEST/1 Model from Performance Data

On the Create BEST/1 Model from Performance Data display, type the following information and press Enter:

- **A:** Enter the description text for your model.
- **B:** Enter the member name containing the iSeries performance data.
- **C:** Enter the name of the library containing the measured iSeries performance data.

Create BEST/1 Model from Performance Data

Model : YOURMODEL

Type choices, press Enter. Use *SLTHOUR to select an hour-long time period or use *SLTITV to select select first and last interval of a one to two hour time period. The time period selected should be representative of your peak processing activity.

Text A

Performance data:

Member B

Library C

Name, F4 for list

Name

Start time *SLTHOUR

Start date *FIRST

Time, *FIRST, *SLTHOUR, *SLTITV

Date, *FIRST

Stop time *LAST

Stop date *LAST

Time, *LAST

Date, *LAST

F3=Exit F4=Prompt F12=Cancel

Figure A-5 Create BEST/1 Model from Performance Data

Note: Use the *Start time* and *date* and *Stop time* and *date* specifications to select the performance data you want to include in your model. Use the F1 (Help) key for assistance on using these fields.

Selecting the time interval

If you select *SLTITV for the start time in the display shown in Figure A-5, a display appears that allows you to select the start time and end time of the performance data to be included in the model (Figure A-6).

Select Hour-long Period

Library : QPFRDOMINO Performance member . . : MEMBER10

Type option, press Enter. Select an hour-long period representative of your peak processing activity. Each time period is the sum of a one-hour performance monitor data collection. The time stamp shown is that of the last interval in the time period.

1=Select

Opt	Date	Start Time	---Transaction--- Count	Rsp Time	--CPU Util-- Total	Inter	I/Os per Sec Sync	Async
	11/23/01	16:28:25	307	.1	42	0	12	58
	11/23/01	16:33:25	201	.5	47	0	10	52

Bottom

F3=Exit F12=Cancel F15=Sort by interval F16=Sort by count
F17=Sort by rsp time F18=Sort by total CPU util F19=Sort by total I/Os

Figure A-6 Select Hour-long Period

Select the start and end times *only* by entering a **1** in the Opt column.

Classify Jobs into workloads

The display in Figure A-7 allows you to specify how you want to classify the resource usage into workloads. Select option **2** and press Enter.

Classify Jobs

Select one of the following:

1. Use default job classification

2. Classify jobs into workloads

3. Use existing job classifications

Selection

2

F3=Exit F12=Cancel

Figure A-7 Classify Jobs

Specify Job Classification Category

The display shown in Figure A-8 allows you to select the job category you want to use in assigning the iSeries Domino server jobs to workloads. Select the category that can assist you in defining your workloads. For example, if each of the Domino server workloads runs in separate subsystems, you may select option 6 (Subsystem) as the classification category. For this example, we assume that the Domino workloads are, in fact, separated into different subsystem. If this is not the case, use Job name as a classification and separate the different jobs such as SERVER, UPDATE, and so on.

Specify Job Classification Category

Type choice, press Enter.

Category 6

1=User ID

2=Job type

3=Job name

4=Account code

5=Job number

6=Subsystem

7=Pool

8=Control unit

9=Comm line

10=Functional area

F3=Exit

F12=Cancel

Figure A-8 Specify Job Classification Category

Edit Job Classifications

The details of the display shown in Figure A-9 depend on your selection of the classification category. The following example is based on the subsystem.

Edit Job Classifications

Enter workload names and category values which are assigned to each workload, press Enter. Jobs with unassigned values become part of workload QDEFAULT.

Workload	Subsystem	Workload	Subsystem	Workload	Subsystem
_____	_____	_____	_____	_____	_____
_____	_____	_____	_____	_____	_____
_____	_____	_____	_____	_____	_____
_____	_____	_____	_____	_____	_____
_____	_____	_____	_____	_____	_____
_____	_____	_____	_____	_____	_____
_____	_____	_____	_____	_____	_____
_____	_____	_____	_____	_____	_____

F3=Exit

F9=Display values from data

F12=Cancel

To display values from performance data, press F9

More...

Figure A-9 Edit Job Classifications

Press F9 to analyze your performance data and present the available values within the category selected.

Assign the iSeries Domino server jobs to as many workloads as you want to create within your model (Figure A-10). If your measured data has interactive work in addition to Domino server applications, you can define these as well.

Note: We recommend that you do not explicitly assign *LIC tasks to a specific workload.

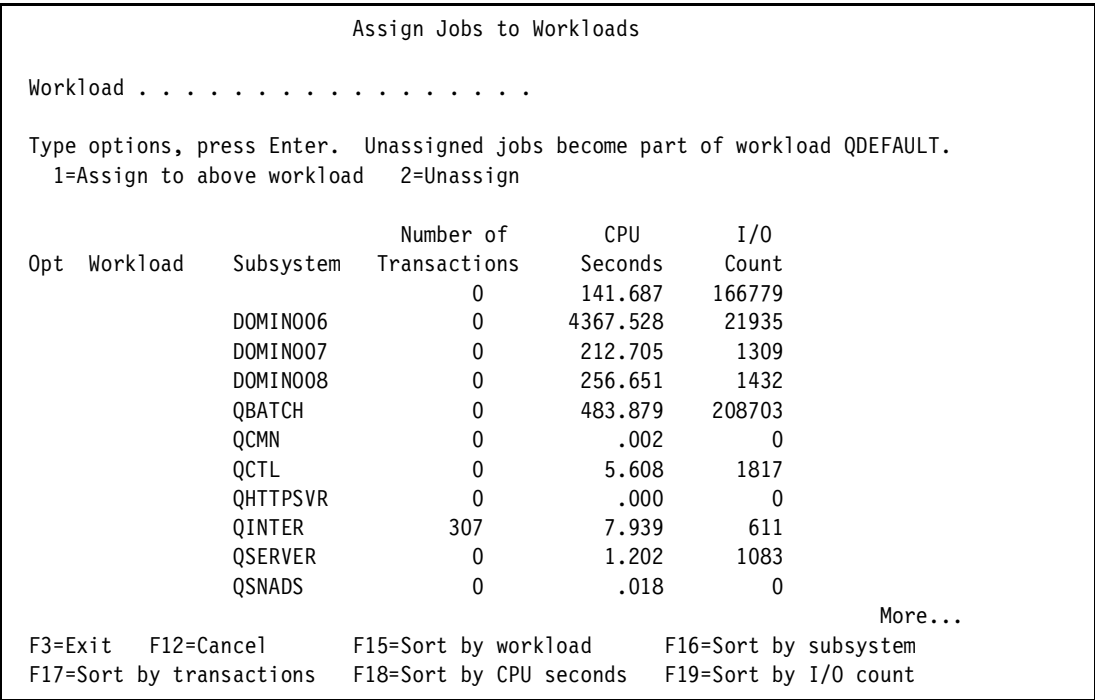


Figure A-10 Assign Jobs to Workloads

BEST/1 has internal algorithms to allocate *LIC tasks to the appropriate workload.

Specify Paging Behaviors

Accept the default value of *GENERIC for all workloads as shown in Figure A-11.

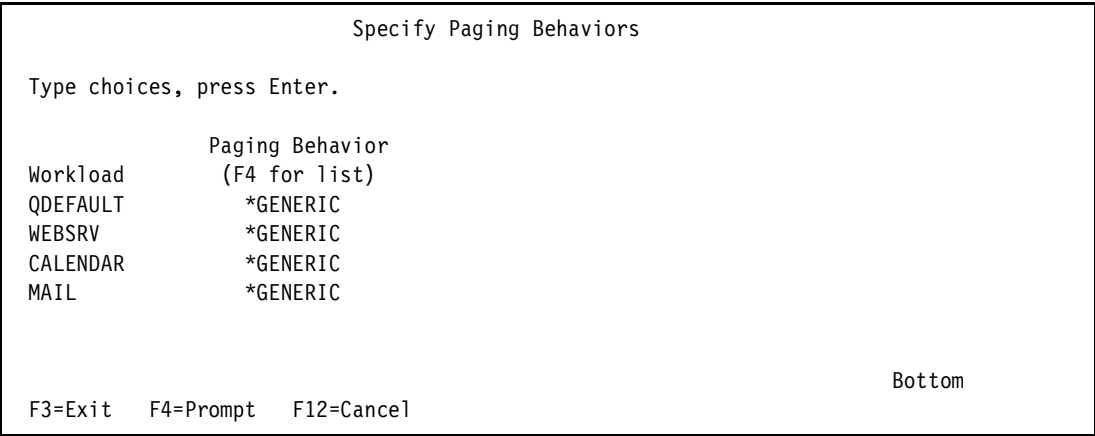


Figure A-11 Specify Paging Behaviors

Define Non-Interactive Transactions

The display in Figure A-12 allows you to specify the number of CPUMs for each server workload. Refer to “Selecting a capacity planning unit of measure” on page 401 for a discussion on the selection of CPUMs.

For CPUMs per Hour, the value entered is per hour regardless of the time period over which the model was built. Ensure that you change the "Type" field to *NONE for all of the server workloads.

Define Non-Interactive Transactions

Job classification category : Subsystem

Type choices, press Enter.

Workload	---Activity Counted as Transaction---	Quantity	Total Transactions when Type = *NONE
QDEFAULT	*LGLIO	100.0	0
WEBSRV	*NONE	100.0	10000
CALENDAR	*NONE	100.0	5000
MAIL	*NONE	100.0	20000

Bottom

Type: *LGLIO, *CMNIO, *CPUSEC, *PRINT, *NONE

F3=Exit F12=Cancel

Figure A-12 Define Non-Interactive Transactions

Save Job Classification Member

The display in Figure A-13 allows you to save job classifications as a member in a library.

Save Job Classification Member

Change values if desired, press Enter.

Member A Name

Library B Name

Text C

Replace N Y=Yes, N=No

F12=Cancel

Figure A-13 Save Job Classification Member

Enter the following information and press Enter:

- ▶ **A:** Enter the member name to contain the classification.
- ▶ **B:** Check the library name for BEST/1 objects.
- ▶ **C:** Enter description text for your classification.

Confirm Creation of BEST/1 Model

Check the model and library name and the associated text and press Enter to submit a batch job to create your BEST/1 model. The display shown in Figure A-14 appears.

Confirm Creation of BEST/1 Model		
Type choices, press Enter.		
Model	A	Name
Library	B	Name
Text	C	
Replace	N	Y=Yes, N=No
Job name	CRTBESTMDL	Name, *JOB
Job description	QPFRJOB	Name, *NONE, *USRPRF
Library	QPFR	Name, *LIBL, *CURLIB
F12=Cancel		
Member JOBCCLASS has been saved		

Figure A-14 Confirm Creation of BEST/1 Model

Enter the following information and press Enter:

- ▶ **A:** Check the member name to contain the classification.
- ▶ **B:** Check the library name for BEST/1 objects.
- ▶ **C:** Check the description text for your classification.

Next, you return to the Work with BEST/1 Models display. You may issue the Work with Submitted Job (WRKSBMJOB) command to find out when your create model job completes, tune the Display Message (DSPMSG) command to look for a model completion message, or repeat using the F5 key (Refresh) until you see your new model name appear on the Work with BEST/1 Models menu. After you see your new model on this display, proceed to the next step.

Validating the BEST/1 model

Before you evaluate the affect of workload growth, review the model created by BEST/1 to ensure that it conforms to the measured workload and response times. This section discusses the process to achieve this.

Model objectives: A BEST/1 model is considered properly calibrated when the measured data values have response times within 0.5 seconds of the corresponding predicted data values, and the resource utilizations between the measured and predicted are within 20% of each other.

In Domino server capacity planning, you can ignore any differences in communications resource utilization because we limit the modeling to the Domino server only.

Refer to *BEST/1 Capacity Planning Tool*, SC41-5341, which discusses the adjustments required when modeling “server jobs”.

You must rely on your experience with BEST/1 and your understanding of the Domino server application and the OS/400 operating environment to effectively adjust or calibrate the model.

BEST/1 workload components

You may have to verify and modify the workloads if analysis shows that BEST/1 calculations for some important resource usages do not coincide with actual measurements. These include:

- ▶ Total CPU utilization
- ▶ Disk I/O activity
- ▶ Number of non-interactive transactions per hour model

Workload

A BEST/1 model is made of one or more user-defined workloads. This is the main unit of input to capacity planning.

From the Work with BEST/1 Model display, select option 1 (Work with workloads) to see all the workloads in a BEST/1 model. The display shown in Figure A-15 appears.

Work with Workloads

Model/Text: YOURMODEL C

Type options, press Enter.

1=Create 2=Change 3=Copy 4>Delete 5=Display 6=Print 7=Rename

8=Save workload to workload member 9=Edit transactions

Opt	Workload	Text
	CALENDAR	Measured from QPFRDOMINO (MEMBER10)
	MAIL	Measured from QPFRDOMINO (MEMBER10)
	QDEFAULT	Measured from QPFRDOMINO (MEMBER10)
	WEBSRV	Measured from QPFRDOMINO (MEMBER10)

Bottom

F3=Exit F6=Add saved workload F9=Add predefined workload F12=Cancel

F13=Combine workloads

Figure A-15 Work with Workloads

Function

Each workload is made up of one or more functions. Each defined workload created using performance data defaults to one function per user per hour. This function represents all of the work done by this workload. Use option 5 to display the functions in a workload (see Figure A-16).

Note: Non-interactive workloads have a usage mode represented by N/A.


```

                                Display Workload

Workload . . . . . : MAIL
Workload text . . . . . : Measured from QPFRDOMINO (MEMBER10)
Workload type . . . . . : *NORMAL
CPU architecture . . . . . : *RISC
Usage mode . . . . . : 4                1=Casual, 2=Interrupted, 3=Steady,
                                         4=N/A

Type option, press Enter.
5=Display

Opt  Function      Function Text      Functions  Avg K/T      Inter Tns
      MAIL          Function of MAIL      per User    (secs)      per Function
                                         1.00        N/A          .00

                                           Bottom

F3=Exit   F9=Display chars to comm line resources   F10=Display I/Os to ASPs
F11=Display non-inter transactions   F12=Cancel

```

Figure A-16 Display Workload

Two additional values are available at the Function level to assist you in calibrating a BEST/1 model. These are:

- ▶ Key/think time (not applicable for non-interactive work)
- ▶ Additional delays

These parameters can be modified by selecting the Change Function option (Figure A-17).

```

Change Function

Workload . . . . . : MAIL          Measured from QPFRDOMINO (MEMBER10)
Function . . . . . : MAIL

Change fields, press Enter.
Function text . . . . . : Function of MAIL
Key/Think time . . . . . : N/A          Seconds
Additional delays . . . . . : .0          Seconds

Transaction      Pool      Transactions      CPU Time      Total
Type            ID        per Function    (Secs)         I/Os
  2              2          1.00           1.513          6.5
  2              5          20000.00       16.521          1.2
  2              1          1.00           11.859         407.3

Bottom

Transaction Type: 1=Interactive, 2=Non-interactive

F3=Exit  F6=Work with transactions  F12=Cancel

```

Figure A-17 Change Function

Transaction

Each function is made up of one or more transactions. The details of a function and the transactions included in the function are displayed using option 5. The transactions in a Domino server workload are non-interactive. The display in Figure A-18 shows the transactions per hour.

Display Function						
Workload	MAIL		Measured from QPFRDOMINO (MEMBER10)			
Function	MAIL		Function of MAIL			
Key/Think time	N/A		Seconds			
Additional delays0		Seconds			
Type options, press Enter.						
5=Display						
Opt	Transaction Type	Pool ID	Priority	Transactions per Function	CPU Time (Secs)	Total I/Os
	2	2	0	1.00	1.513	6.5
	2	5	20	20000.00	16.521	1.2
	2	1	0	1.00	11.859	407.3
						Bottom
Transaction Type: 1=Interactive, 2=Non-interactive						
F3=Exit F12=Cancel						

Figure A-18 Display Function

The display in Figure A-19 shows an example of the information included in a transaction, which is obtained by selecting option 5.

Display Transaction			
Workload	MAIL	Measured from QPFRDOMINO (MEMBER10)	
Function	MAIL	Function of MAIL	
Transaction Type	2	1=Inter, 2=Non	
Pool ID	5		
CPU Priority	20		
CPU time	16.521	Secs (on B10)	
Permanent writes	98.2	Percent	
Chars transferred in	0		
Chars transferred out	0		
Exceptional wait0	Msec	
Paging behavior	*GENERIC		
		Reads	Writes
Sync DB I/Os0	.0	
Async DB I/Os0	.0	
Sync non-DB I/Os1	1.1	
Async non-DB I/Os0	.0	
Press Enter to continue.			
F3=Exit F12=Cancel F13=Display paging behavior			

Figure A-19 Display Transaction

Note: You can make adjustments to the model for any or all of the following components:

- ▶ Workload
- ▶ Function
- ▶ Transaction

Work with BEST/1 Models

On the Work with BEST/1 Models menu (Figure A-3 on page 407), select option **5** (Work with) for your model.

Analyzing the current model

On the Work with BEST/1 Model menu (Figure A-20), select option **5** (Analyze current model). This option runs the specified workload against the current configuration in the model.

Work with BEST/1 Model

Performance data . . . : QPFRDOMINO (MEMBER10)
Model/Text : YOURMODEL Domino Workload

Select one of the following:

1. Work with workloads

2. Specify objectives and active jobs

5. Analyze current model

6. Analyze current model and give recommendations

7. Specify workload growth and analyze model

10. Configuration menu

11. Work with results

More...

Selection or command

==> 5

F3=Exit F4=Prompt F9=Retrieve F12=Cancel F15=Save current model
F17=Analyze using ANZBESTMDL F22=Calibrate model F24=More keys

Figure A-20 Work with BEST/1 Model

Work with Results

Use the facilities within this function (see Figure A-21) to evaluate the suitability of the BEST/1 model. The values predicted by the model may be substantially different from the values measured by Collection Services and the application profile measurements. In this case, you must make the necessary "artistic adjustments" to the model.

Calibration expertise is where skill and experience play a major role.

Work with Results

Printed report text Domino workload

Type options, press Enter.
5=Display 6=Print

Opt	Report Name
	Measured and Predicted Comparison
	Analysis Summary
	Recommendations
	Workload Report
	ASP and Disk Arm Report
	Disk Resources Report
	Main Storage Pool Report
	Communications Resources Report
	All of the above

Bottom

F3=Exit F12=Cancel F14=Select saved results F15=Save current results
F18=Graph current results F19=Append saved results F24=More keys

Figure A-21 Work with Results

Some of the possible anomalies are discussed in the following section.

Workload type **BATCHJOB*

On the Work with Results menu (Figure A-21), you may see the message Current system CPU cannot handle the measured workload. This observation may not coincide with the analysis of iSeries performance data.

On the Work with Results menu, type option 5 (Measured and predicted comparison). This shows a column of measured statistics compared to the values evaluated by the BEST/1 model.

The example in Figure A-22 shows the CPU as a highlighted field, which indicates that CPU usage is in excess of 98%. Relatively high predictions for other system resources may also exist.

Measured and Predicted Comparison		
	Measured	Predicted
Total CPU util :	42.2	42.2
Disk IOP util :	.0	.0
Disk arm util :	5.8	11.9
Disk IOs per second :	192.4	186.2
Multifunction IOP util :	10.9	10.9
Disk IOA util :	9.2	17.4
LAN IOA util :		.0
WAN IOA util :		.0
Integrated PC Server IOA util :	.0	.0
More...		
Interactive:		
CPU util :	.1	.1
Int rsp time (seconds) :	.1	.0
Transactions per hour :	335	334
Non-interactive thrupt :	36520	36765
Performance estimates -- Press help to see disclaimer.		
F3=Exit F6=Print F9=Work with spooled files F12=Cancel		
F17=Calibrate response time		

Figure A-22 Measured and Predicted Comparison

It is important to understand how BEST/1 came up with the predicted CPU utilization of approximately 98%. This can affect the modeling of all Domino "server jobs". This happens because they are not perceived by the iSeries performance measurements as interacting with any external agent and, therefore, are treated as a batch job.

For Modeling Batch Jobs, BEST/1 assigns a workload type of *BATCHJOB when all jobs in the BEST/1 workload have met *all* of the following criteria:

- ▶ All jobs are non-interactive jobs.
- ▶ All jobs were active for at least 95% of the time in which Collection Services collected data.
- ▶ All storage pools used by the jobs service only non-interactive jobs.

The *BATCHJOB workload type was designed to assist in modeling traditional batch job runtime, which gives all additional CPU capabilities to a *BATCHJOB workload.

However, this algorithm has problems when the actual non-interactive environment contains Domino server-type jobs that periodically wait for work, similar to an interactive workstation job. Domino server jobs include mail serving, Web serving, and calendar work that can all be considered interactive in nature.

If the workload is identified by BEST/1 as *BATCHJOB, BEST/1 assigns all available CPU that is not used by other workloads to the server workloads, such as Domino. This causes high-predicted CPU utilization.

The BEST/1 user must understand the Domino server job implementation included in performance data and manually change the workload attribute from *BATCHJOB to *NORMAL if it shows as *BATCHLOAD. Making this change and other changes to the model are called *calibrating the model*.

Transactions per function

Ensure that the number of transactions per function match the value input to the model.

When Adjusting Transactions, BEST/1 always takes some of the unassigned OS/400 system work (pool 2) and Licensed Internal Code (LIC) work (pool 1) and adds a portion to all BEST/1 workloads. It does this because some system and LIC work is required to do tasks such as manage the job and perform disk and communications I/O operations. For non-interactive workloads that are classified as *BATCHJOB, the number of active jobs is set to the number of jobs that contribute to the workload. This may not reflect the true profile of the workload. If the workload is made up of non-interactive jobs but are *not* classified as *BATCHJOB, the number of active jobs is set to 1.

For example, you may have six active jobs, but the workload was really derived from only four. You may have to examine the number of Active Jobs shown in the display in Figure A-23 by selecting option 2 from the Work with BEST/1 Model menu.

Specify Objectives and Active Jobs						
Model/Text: YOURMODEL C						
Type changes, press Enter.						
Workload	Connect	Workload Type	Active Jobs	----Interactive----- Rsp Time	Thruput	Non-inter Thruput
CALENDAR	*LOCAL	*NORMAL	1.0	.0	0	0
MAIL	*LOCAL	*NORMAL	1.0	.0	0	0
QDEFAULT	*LOCAL	*NORMAL	.1	.0	0	0
QDEFAULT	*LAN	*NORMAL	4.0	.0	0	0
WEBSRV	*LOCAL	*NORMAL	1.0	.0	0	0
						Bottom
F3=Exit F11=Show all quantities F12=Cancel F15=Sort by connect type						
F19=Work with workloads						

Figure A-23 Specify Objectives and Active Jobs

This difference can result in erroneous reporting of the number of non-interactive transactions for the function.

Manual calibration of resource usage

BEST/1 allows you to manually calibrate resource utilization, such as CPU and disk, by adjusting workload functions until the predicted utilizations are a close approximation to the measured values. Manual calibration requires a good understanding of the parameters that you are changing.

For model calibration, press the F22 key to enter the manual calibration mode. While you are in manual calibration mode, any changes to the number of active jobs, or functions per user do not affect total workload throughput (transactions per hour). The values for transactions per function are modified to offset changes to either active jobs or functions per user. In addition, all paging coefficients are recalculated during analysis.

After manual calibration, return to the Work with BEST/1 Model menu, and press F22 to exit the manual calibration mode.

Calibrating response times

Domino server applications are reported as non-interactive jobs, and any external interfaces are not recognized. Therefore, BEST/1 does not recognize the impact of any communications delays or overhead.

However, if you need to represent an "interactive response time" to a Domino server application (because the user perceives performance from this viewpoint), you can adjust the BEST/1 representation of response time by adjusting the Additional delays value in the workload specification. The response time indication visible in the Display Workload Report display should be considered as representing internal response time.

For example, on the Work with Results menu, select option 5 to display the workload report. You can then display the response time details using the F11 key. Add an amount equal to the difference between the perceived user response time and the internal response time shown in the Workload Report. See Figure A-24.

Display Workload Report								
Period: Analysis			Total	-----Rsp Time Secs spent in-----				
Workload	Type	Connect	Rsp Time	CPU	I/O	Pool	Comm	Other
QDEFAULT	1	*LOCAL	.3	.0	.0	.0	.2	.0
QDEFAULT	1	*LAN	20.9	.0	.0	.0	20.9	.0
CALENDAR	2	*LOCAL	.0	.0	.0	.0	.0	.0
MAIL	2	*LOCAL	.2	.2	.0	.0	.0	.0
QDEFAULT	2	*LOCAL	1.0	.5	.5	.0	.0	.0
QDEFAULT	2	*LAN	1.0	.5	.5	.0	.0	.0
WEBSRV	2	*LOCAL	.0	.0	.0	.0	.0	.0
								Bottom
Type: 1=Interactive, 2=Non-interactive, 3=*BATCHJOB								
Performance estimates -- Press help to see disclaimer.								
F3=Exit			F10=Re-analyze			F11=Workload summary		
F15=Configuration menu			F17=Analyze multiple points			F12=Cancel		
						F24=More keys		

Figure A-24 Display Workload Report

Return to the Work with BEST/1 Model menu and select option 1 (Work with workloads). Select option 2 (Change) for the workload. Then you see the display in Figure A-25. Press F6 (Work with functions), and select option 2 (Change) for the workload. Increase the additional delay.

Change Workload

Workload : MAIL

CPU architecture : *RISC

Type changes, press Enter.

Workload text Measured from QPFRDOMINO (MEMBER10)

Workload type *NORMAL F4 for list

Usage mode 4 1=Casual, 2=Interrupted, 3=Steady, 4=N/A

Function Text	Functions per User	Avg K/T (secs)	-----Tns per Function----- Inter	Non-inter
Function of MAIL	1.00	N/A	.00	20002.00

Bottom

F3=Exit F4=Prompt F6=Work with functions F9=Specify chars to comm lines

F10=Specify I/Os to ASPs F11=Show all functions F12=Cancel

Figure A-25 Change Workload

Exceptional wait time can also be added at the transaction level of the workload (Figure A-26).

Display Transaction

Workload : MAIL

Function : MAIL

Measured from QPFRDOMINO (MEMBER10)

Function of MAIL

Transaction Type : 2 1=Inter, 2=Non

Pool ID : 5

CPU Priority : 20

CPU time : 16.521 Secs (on B10)

Permanent writes : 98.2 Percent

Chars transferred in : 0

Chars transferred out : 0

Exceptional wait : .0 Msec

Paging behavior : *GENERIC

	Reads	Writes
Sync DB I/Os0	.0
Async DB I/Os0	.0
Sync non-DB I/Os1	1.1
Async non-DB I/Os0	.0

Press Enter to continue.

F3=Exit F12=Cancel F13=Display paging behavior

Figure A-26 Display Transaction

Saving the BEST/1 model

When calibration of the BEST/1 model is complete, save the model and workloads to use it later for modeling the growth of this workload and for modeling it with other measured data workloads. This is especially important if considerable manual adjustments are made to the model.

Perform the following tasks:

1. Return to the Work with BEST/1 Model menu. Select option **1** (Work with workloads).
2. Select option **8** (Save workload to workload member) for each of the workloads and save the workload.

Save Workload to Workload Member		
Change values if desired, press Enter.		
Member	A	Name
Library	B	Name
Text	C	
Replace	N	Y=Yes, N=No
CPU architecture	*RISC	*CISC, *RISC
F12=Cancel		

Figure A-27 Save Workload to Workload Member

Enter the following information and press Enter:

- **A:** Enter a member name to contain the workload.
- **B:** Enter the library name for BEST/1 objects.
- **C:** Enter a description text for your workload.

The workload member is saved to a file called QACYWKLS in your library. You can use this workload in any other system, or you can add this workload to any other model from any other system.

On the Work with BEST/1 Model menu, select F15 (Save the current model), and enter a name (Member) of the model. Press Enter to save the model.

Work with Selected BEST/1 Model

Select option **7** to specify the workload growth amounts and the number of periods you want the model to grow.

Specifying workload growth and analyzing the model

The display shown in Figure A-28 allows you to grow all of the workloads by a single percentage for up to 10 periods.

Specify Growth of Workload Activity

Type information, press Enter to analyze model.

Determine new configuration Y Y=Yes, N=No

Periods to analyze 5 1 - 10

Period 1	Period 1 Name
Period 2	Period 2 Name
Period 3	Period 3 Name
Period 4	Period 4 Name
Period 5	Period 5 Name

-----Percent Change in Workload Activity-----

Workload	Period 1	Period 2	Period 3	Period 4	Period 5
*ALL	.0	20.0	20.0	20.0	20.0

Bottom

F3=Exit F11=Specify growth by workload F12=Cancel
 F13=Display periods 6 to 10 F17=Analyze using ANZBESTMDL

Figure A-28 Specify Growth of Workload Activity

Pressing F11 allows you to specify growth rates for each workload in the model (Figure A-29).

Specify Growth of Workload Activity

Type information, press Enter to analyze model.

Determine new configuration Y Y=Yes, N=No

Periods to analyze 5 1 - 10

Period 1	Period 1 Name
Period 2	Period 2 Name
Period 3	Period 3 Name
Period 4	Period 4 Name
Period 5	Period 5 Name

-----Percent Change in Workload Activity-----

Workload	Period 1	Period 2	Period 3	Period 4	Period 5
CALENDAR	.0	10.0	15.0	20.0	15.0
MAIL	.0	10.0	20.0	25.0	10.0
QDEFAULT	.0	20.0	20.0	20.0	20.0
WEBSRV	.0	15.0	20.0	25.0	10.0

Bottom

F3=Exit F11=Specify total growth F12=Cancel F13=Display periods 6 to 10
 F17=Analyze using ANZBESTMDL

Figure A-29 Specify Growth of Workload Activity by workload

You can either have BEST/1 determine the required upgrade system, or you can modify the configuration and review the effect on system resource usage as the workload increases. Status messages are issued as growth analysis is performed. When analysis is complete, the message Model has been analyzed appears.



SMTP and AnyMail/400 Mail Server Framework (MSF) jobs

The Domino server jobs, which are described in the following sections, appear in the Domino subsystem. This jobs are more representative of the SMTP function sets within Domino for iSeries, when SMTP services are configured with *MSF to support both Domino mail and iSeries mail. This support uses the iSeries SMTP server in combination with the AnyMail/400 Mail Server Framework.

These jobs are highlighted with the intent that if your system should represent a high utilization of any of these jobs, you have more knowledgeable information in front of you to determine just why this is happening. Some of these jobs will appear just briefly, and others may stay active during extended periods.

For more information about SMTP and AnyMail/400, see the redbook *Lotus Domino for AS/400 Internet Mail and More*, SG24-5990.

CCCVS

SMTP Local Deliver Request Queue. Conversion Services.

DRT

The Delivery Report Task (DRT) is responsible for collecting the statistics. Therefore, if the DRT is not running, statistics are not updated. This also means that the statistics will not be updated at the point when every message is processed. Statistics are only updated when DRT processes the messages in the queues. The Total Routed and Waiting Recipients statistics will not be incremented.

The TotalKBTransferred is an approximation. Each message is rounded to the closest KB size and this value is added to the total.

The Waiting statistic is first a snapshot of the inbound work queue count. Then, the outbound work queue count is added to the inbound. In addition, if either queue is greater than 100, the count for that queue will only be 100.

Examples of statistical information reported include:

- ▶ The total number of messages sent and received by the MTA since the MTA started (MTA.Smtp.Transferred)
- ▶ The total accumulated size of all messages in KB sent and received by the MTA since the MTA started (MTA.Smtp.TotalKBTransferred)
- ▶ The total number of Non-Delivery Reports and Undeliverable Message Notifications generated by the MTA since the MTA started (MTA.Smtp.TransferFailures)
- ▶ A snapshot of the total number of messages pending in the MTA internal work queues (MTA.Smtp.Waiting)
- ▶ A snapshot of the total number of messages that are marked dead in the MTA internal work queues (MTA.Smtp.Dead)

IMSGCNV

The Inbound Message Conversion Task (task name SMTPMTA MSGCNV) converts messages received by the Inbound Sessions Handlers to a Notes format. It also converts the destination user address to a Notes format and checks that this address is deliverable.

If the message is not convertible or the address not deliverable, the Inbound Message Conversion Task indicates that the message has failed delivery so that a Non-delivery Report can be generated. If the message is not destined for Notes, it puts the message into the work queue for the Outbound Transport. The process flow is shown in Figure B-1.

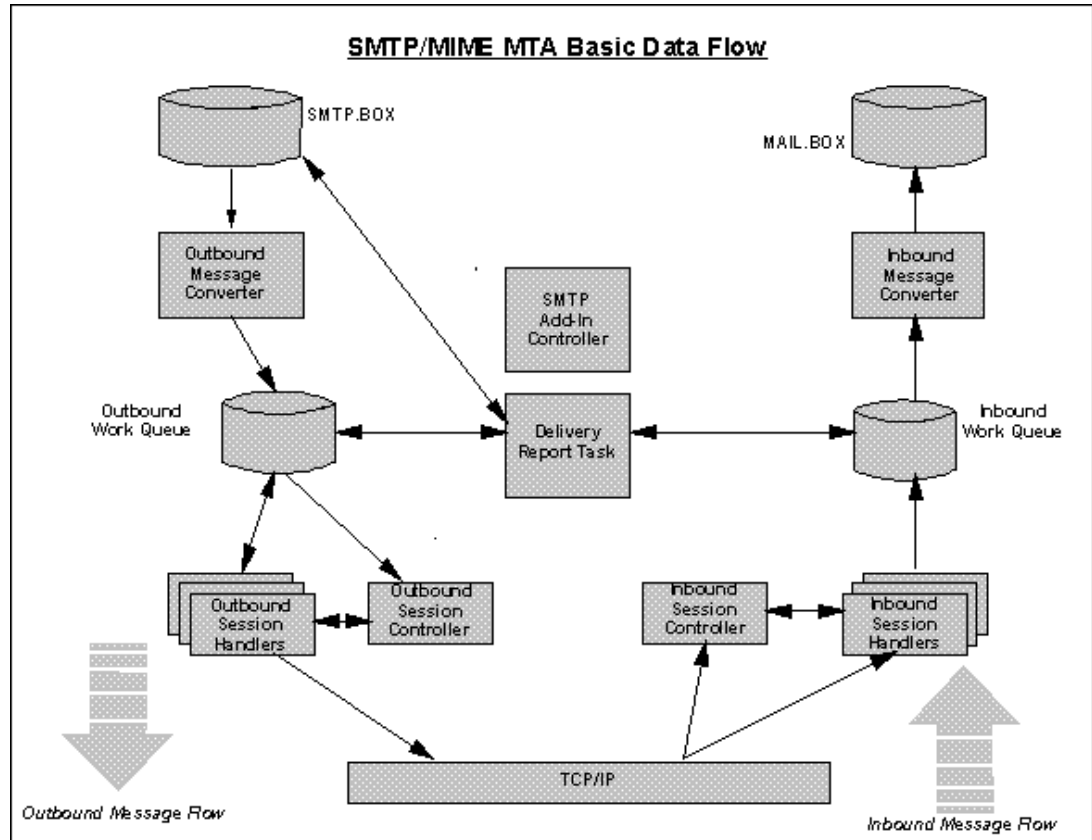


Figure B-1 Outbound/inbound message converter flow

ISESCTL

This task is started by the SMTP add-in task when the MTA loads. The Inbound Session Controller appears on the server console as SMTPMTA ISESCTL listening on SMTP port 25. This controller listens for connection requests on TCP port 25. When a connection request is made, the controller checks an internal list for available session handlers, transfers the connection to such a handler, disconnects, and returns to its listening state. If no handler is available, the controller spawns a new handler, provided that the maximum number of inbound handlers has not been reached.

If the maximum number of inbound handlers has been reached (as defined in the Internet Message Transfer Agent section of the Server document), the controller stops listening on TCP port 25. This state of "not listening" is very infrequent. However, when it occurs, the controller relies on the standard TCP retry time-out to ensure that the request is eventually handled.

Note: Some IP stack implementations queue a certain amount of session requests when all session handlers are busy. In such cases, all session handlers and the session queue have to be busy or full before a connection fails.

Potential performance impact

On low end iSeries servers that are dedicated as a Lotus Notes server, the SMTP MTA server job, ISESCTL, may use excessive amounts of CPU (approximately 40%) that prevent other jobs from starting or acquiring CPU to perform tasks. The error may also occur on high-end systems, which may be dedicated as a Lotus Notes server. The problem has not been identified to exist on high-end systems. However, there is an indication that the problem does occur, but it is not recognized.

Suggested performance tuning

This issue was reported to Lotus Quality Engineering and has been addressed in Domino for iSeries QMUs 4.6b.03 and 4.6.2a.

Review this excerpt from the Lotus Domino Release 4.6b.03 and 4.6.2a QMU fix lists:

“SPR #JBLM3XKULB - ISESCTL is using approximately 40% of CPU UTIL causing processor utilization to max out.”

Domino 4.6b.03 and 4.6.2a have been changed to prevent the SMTP MTA ISESCTL job from obtaining excessive amounts of CPU for extended periods.

ISESHLR

This task is spawned by the Inbound Session Controller on an as-needed basis and appears on the server console as SMTPMTA ISESHLR0. The handler "speaks" SMTP (as defined by RFC 821) with connected hosts that have been passed to the handler by the session controller. Information received during an SMTP session is placed into a document in the Inbound Work Queue (the Notes database that temporarily stores messages that need conversion to Notes format) for processing by the conversion task.

When the message is complete, the handler checks the state of the Inbound Message Conversion task (which converts messages from RFC-standard Internet format to Notes format). If the task is idle, the handler "kick starts" the task through Inter-Process Communication (IPC). The handler then updates its state information in the internal list of inbound handlers and returns to an available state.

MTA

MTA platform independent and platform dependent OS Interfaces

MTS

Mapping Table Service

MTAWQ

MTA work queue interfaces

OMSGCNV

The Outbound Message Conversion task converts outbound messages from a Notes format to an RFC-standard Internet format. The format of the SMTP message and the conversion of Notes addresses to SMTP Addresses is fully configurable.

OSECTL

This task is also started by the SMTP add-in task during the SMTP initialization and appears on the server console as SMTPMTA OSECTL. This controller waits for notification that there are new messages to send from the Outbound Message Converter (the task that converts outbound messages from Notes format to RFC-standard Internet format). When the controller receives such notification, it checks an internal list for available outbound session handlers and passes the available handler a reference ID to the pending message.

If there are no available handlers, the controller spawns a new handler, provided the maximum number of handlers has not been reached. If the maximum number of outbound handlers has been reached (as defined in the Internet Message Transfer Agent section of the Server document), the controller keeps checking its internal list of handlers until a handler becomes available and then passes the handler a reference ID to the pending message.

Note: As a secondary method for locating new messages to be sent, the Outbound Session Controller routinely polls the Outbound Work Queue for new messages.

OSESHLR

This task is spawned by the Outbound Session Controller on an as-needed basis and appears on the server console as SMTPMTA OSESHLR0. The handler transfers messages to destination host systems.

As new messages appear in the Outbound Work Queue (the Notes database that temporarily stores messages ready for transmission), the Outbound Session Controller passes a message reference ID to an available, or newly spawned, session handler. The handler is responsible for domain name resolution and makes the connection with the destination host system of the message recipient.

Once an SMTP connection is established, the message is transferred according to SMTP commands defined in RFC 821. When the handler has completed a session, it updates the status of the message in the work queue, updates its own state in the internal list of handlers, and returns to an available state.

Note: OSESHLR0 and ISESHLR0 represent the first handlers spawned by their respective controllers. As more handlers are spawned, the number appended to the end of the task name increases. For example, the second Outbound Session Handler spawned appears on the server console as SMTPMTA OSESHLR1.

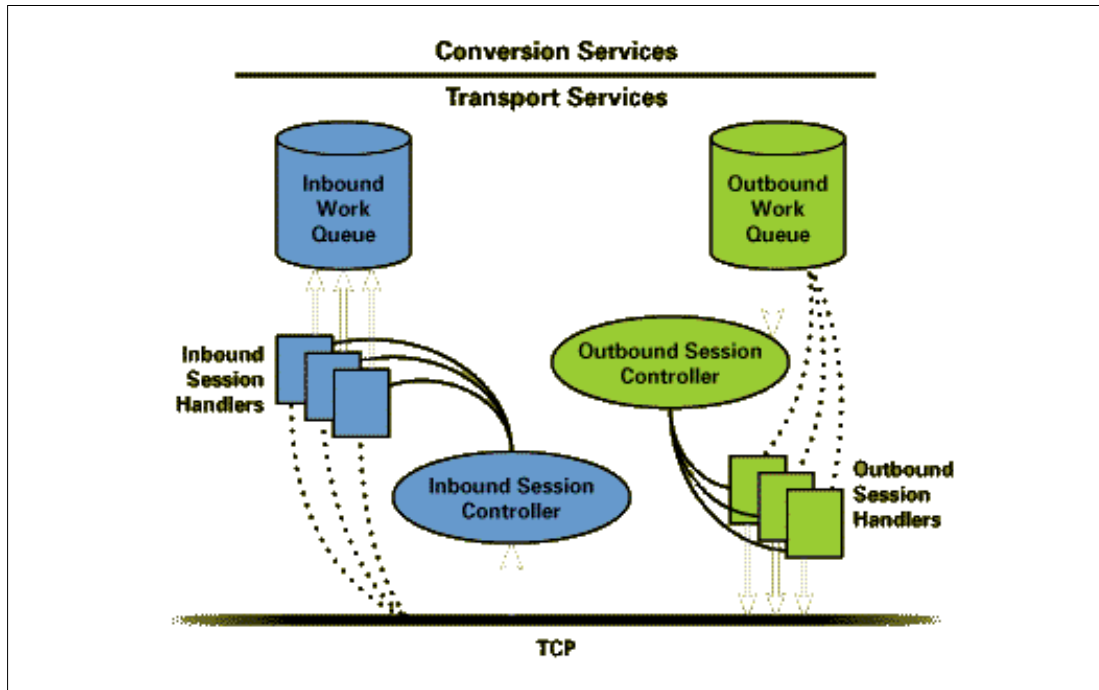


Figure B-2 Message, Transport Services components

SMTPMTA

The Add-in Controller (task name SMTPMTA) is the task that is loaded on the Domino server. It acts as a control point for all the other tasks. All commands for the MTA are sent to the Add-in Controller and it notifies the necessary child processes, for example, Tell SMTPMTA Quit.

SMTPLDQ

SMTP local deliver request queue

Domino for iSeries server programs QNNDxxxx and QSYSWRK

The following programs may use many resources. They are listed here only so you can identify them. There is no way to influence or to tune them. Some of these programs may appear only briefly, while others may stay active during an extended period of time.

Most of the QNNDxxxx programs handles the directory synchronization services. Directory synchronization allows you to keep the iSeries server distribution directory synchronized with the Domino Directory (also known as the Public Address Book or the Name and Address Book)

The QNNDxxxx programs provide the following functions:

- QINSTAPP** Domino server installation program
- QNNDI2NA** Frontend to Notes Domino Directory to SDD (System Database Directory) processor
- QNNDI2SD** Frontend to SDD (System Database Directory) to Notes Domino Directory
- QNNDICFG** Native Notes dir sync main configuration
- QNNDICIN** Native Notes dir sync common code
- QNNDICLN** Frontend directory cleanup program
- QNNDICSD** Cleanup SDD NoteID field
- QNNDIDFV** Native Notes dir sync HiTest Interfaces

QNNDIEND End directory sync jobs
QNNDIENV Sets up Domino application environment
QNNDIFMG Frontend to message queue processor
QNNDIHK Native Notes dir sync DB Hook Driver
QNNDILCK Native Notes dir sync database locking PGM
QNNDILOG Native Notes dir sync configuration logging
QNNDIMAL Native Notes dir sync mail related information
QNNDIMAP Native Notes dir sync mappings configuration
QNNDIMPC Native Notes dir sync SDD—>Domino Directory map tool
QNNDIMSG Native Notes dir sync message handler
QNNDIOBJ Native Notes dir sync iSeries object function
QNNDIPNA Populate Domino Directory to SDD
QNNDIPOP Directory sync populate program
QNNDIPSD Populate SDD to Domino Directory
QNNDIQ2N Native Notes dir sync from SDD to Domino Directory
QNNDIQ2S Native Notes dir sync from Domino Directory to SDD
QNNDIRML Native Notes dir sync mail field handler
QNNDIRQC QNNDIRQC
QNNDIRQS Data queue front-end program
QNNDIRSY Native Notes dir sync common Q processes
QNNDISCD Schedule directory sync "populate"
QNNDISCH SDD search
QNNDISTJ Start queue processor jobs
QNNDISUP SDD update
QNNDIXIT SDD exit program for Notes Dir Sync
QNNDI2NA Front-end to SDD to Domino Directory processor
QNNDI2SD Front-end to Domino Directory to SDD processor
QNNINADD AddIn task
QNNINAPI Native Notes partitioned server API function
QNNINCPP CFGDOMSVR CPP program
QNNINCSD Console display
QNNINCSL Domino console support
QNNINCSS Console start
QNNINEDS End Domino server
QNNINMTA Install MTA
QNNINSDS Start Domino server
QNNINSMT SMTP/MTA program
QNNINSTS Start the server
QNNINSVC Common choices program
QNNINVAL CFGDOMSVR validity checker program
QNNINVCP Common validity checker program



Important notes.ini parameters

This appendix lists in functional groups the some of the most important parameters in the notes.ini file:

- ▶ Table C-1 on page 436 contains the settings in the notes.ini file that are most useful for performance analyzing and tuning.
- ▶ Table C-2 on page 442 shows all the notes.ini logging parameters that are useful for debugging but have a considerable performance cost and should be left switched *off* when not required.
- ▶ Table C-3 on page 444 lists the key notes.ini parameters related to agents that affect performance.
- ▶ Table C-4 on page 446 shows all the Domino notes.ini parameters related to Indexing and full text (FT) indexing, all of which can and will have a significant effect on Domino performance.
- ▶ Table C-5 on page 449 shows the mail related notes.ini parameters that may affect performance.
- ▶ Table C-6 on page 451 shows all the clustering parameters that will affect performance.

Steps for changing the notes.ini file

There are multiple ways that you can edit the notes.ini file. We outline the three most common ways here:

- ▶ Add to the notes.ini section of a configuration document in the Domino Directory (names.nsf - previously called Public Address Book). If you create a default configuration document, you can propagate these changes throughout your domain simply by replicating the Domino Directory.
- ▶ Simply select option 13 from the Work with Domino Servers (WRKDOMSVR) command to edit the notes.ini file for a particular server. This is much more convenient in that you do not have to know or specify the data directory path to the notes.ini file.
- ▶ Through the Operations Navigator graphical user interface to the iSeries, you can work with the Domino servers on a given iSeries server. By right-clicking the server of choice, you can edit the notes.ini file. This is probably the easiest method of editing the notes.ini.

Tips on editing or adding entries in the notes.ini file

You can place notes.ini variables anywhere in the notes.ini file. However, we recommend that you group them near the end of the file to make it easier to find your changes in the future. Here are some tips:

- ▶ Add them before the last entry in the notes.ini file to avoid potential problems with an incorrect final character in the file. If the final character is not character, the server might not recognize the final entry in the file.
- ▶ It is always a good idea to put in a comment before entering a change to the notes.ini file so you have a good record of the change that was made, why, and when the update occurred.
- ▶ For the server to pick up most changes in the notes.ini file, the server needs to be stopped and restarted after the changes are made.

Description of notes.ini variables

Table C-1 contains the settings in the notes.ini file that are most useful for performance analyzing and tuning.

Table C-1 Notes.ini parameters affecting performance

Domino system parameters	Definition
CleanupTimeout	R5.0.7: Allows changes to the default time of 5 minutes allowed for CleanupScriptPath to execute.
DbDir_Refresh_Interval=<minutes>	Reduce database directory scan frequency. Domino periodically scans your data directory for new files or files that have been removed. By default, this scan happens every 15 minutes. If you have a lot of files in your data directory, or a complicated directory structure, traversing the directory can take a significant amount of CPU. You can reduce the overhead by increasing the refresh interval. Keep in mind, however, that it will take Domino longer to recognize new files. Default=15 (minutes) 0=Disabled (Never scan).

Domino system parameters	Definition
EnableJavaAgents	Disables Java Agents when = 0 If the AS/400 Developer Kit for Java (5722-JV1) is installed on your iSeries, the Domino Agent Manager (AMgr) and HTTP server automatically support running Java agents. You can disable Java agents by adding the following line to the notes.ini file for the Domino server: Default=1 ENABLED (if Java installed) See also "EnableJavaAgents" on page 225.
Fixup_Tasks	The maximum number of fixup tasks that are allowed to be started upon server startup to look at databases that need fixing. Default = 2 x the number of CPUs
Fixup=	Controls what database fixup options are run at server startup. We recommend that you do not use this option, especially with transaction logging enabled. (Need fixup -J or serious damage occurs). The Fixup task should be used as a "last resort" when a database cannot be opened. Domino R5 is more robust. Run COMPACT, or UPALL -R on a database, or create a new replica or a database copy before resorting to running Fixup. Only run Fixup on a database that cannot be opened. Fixup deletes any documents that it determines are damaged.
Memory_Quota	OS/2 only will be ignored by Domino for iSeries.
NFS_Buffer_Pool_Size NFS_Buffer_Pool_Size_MB	The NSF_Buffer_Pool_Size (bytes) and NSF_Buffer_Pool_Size_MB specify the maximum size of the NSF buffer pool. We recommend that you set this and PercentAvailSysResources. On iSeries, the default is added to notes.ini as 300 Mb during installation. Without this parameter, the default is 3/8 of total physical memory. For full details, see 7.14.1, "NSF_Buffer_Pool_Size or Nsf_Buffer_Pool_Size_mb" on page 211.
NPN=	Notes Partition Number on iSeries (Domino Partitions). Set by configuration tools and the API; do not adjust by hand!
NOTESPARTITION=1	Defines the notes partition number for this instance.
NSF_DbCache_Maxentries = <#-of-DBs>	Sets the max number of databases that can be cached in the DBCache. Increasing the database cache size may well improve system performance, but may require additional memory and will require Maxentries to be increased. The minimum is 25 and the maximum is 2000 (platform). Default: None, but if this setting is omitted, the number is either 25 or the NSF_Buffer_Pool_Size value is divided by 300K (whichever is greater). See the details in "NSF_DbCache_Maxentries" on page 221.
PercentAvailSysResources	Since R5.0.4, this new parameter controls the amount of memory allocated to each Domino Server (Subsystem) by percentage. With the PercentAvailSysResources variable, you assign a portion (%) of total system memory to each Domino server by specifying a value from 2% to 100%, which represents an absolute percentage of the system's total physical memory. This in turn limits NSF_Buffer_Pool_Size (not present in notes.ini) to 3/8 of the memory allocated by PercentAvailSysResources, leaving 5/8 for other Domino buffers to maintain proportion and balance. Recommendation: Set both PercentAvailSysResources and NSF_Buffer_Pool_Size parameters. See "PercentAvailSysResources" on page 213.
Previous_TransLog_Path	Describes the path to transaction logs in the previous configuration. Allows system to restart after manual changes to logfile status, location or style or loss of log files. Note: Some operating systems require a trailing path separator. Use just for recovery and emergencies only. Default=DATA\LOGDIR

Domino system parameters	Definition
Previous_TransLog_Status	Describes the status of transaction logging in the previous configuration. Allows system to restart after manual changes to logfile status, location or style or loss of log files. Use just for recovery and emergencies only. Default=0 (DISABLED)
Previous_TransLog_Style	Logging style in the previous configuration DEFAULT=0 (Circular) Allows the system to restart after manual changes to logfile status, location or style or loss of log files. Use for recovery and emergencies only. Default=0 (CIRCULAR)
Replicators=	Specifies the number of Replicator tasks that can run concurrently on the server. A replicator task synchronizes a source database with its replicas. Starting more than one replicator allows a server to replicate its databases with more than one server simultaneously. If, however, you enable two replicators, and the server only needs to replicate with one other server, then only one replicator is used. The other task remains idle. Multiple replicators will speed up synchronization especially in clustering, but increase workload significantly on the server. Default=1 Recommendation: (number of processors -1) Need to restart the server or start extra replicators with the load replica console command.
Server_Max_Concurrent_Trans	Regulates the number of concurrent transactions that can occur in a server at one time. When the number of concurrent transactions is reached, the other transactions are put into a wait state until one of the active transactions completes. Note: The R4 value of -1 must not be used in R5 because Domino now optimally uses thread pools. These pools are set by default to 40 threads per port. If you set this parameter (default 20), be aware that IOCP threads will default to two times this number (for example, 40). A value of -1 means unlimited and will set IOCP threads to twice unlimited and causes very serious performance problems with threads. Default=20 Recommendation: Do not use. For further details, see "Server_Max_Concurrent_Trans" on page 216.
Server_MaxSessions	Defines the number of user sessions a Domino server can accommodate. When a new user attempts to logon, if the current number of sessions is greater than the value of Server_MaxSessions, Domino closes the least-recently-used session. In order for a session to be considered for closing, it must have been inactive for at least one minute. If there are too many concurrent (active) sessions on the Domino server, errors may occur. Reaching this limit on iSeries causes the SERVER job to take up a lot of CPU with a single thread. Users see the message "remote system no longer responding". Default=Not applicable For further details, see "Server_MaxSessions" on page 217.
Server_MaxUsers	Specifies the maximum number of active NRPC user sessions allowed on a server. This is <i>only</i> effective for NRPC (Notes Client) users and does not affect HTTP users or sessions. When this number is reached, the server state becomes "MaxUsers", and the server stops accepting new Database Open requests from NRPC users. This prevents not only new sessions from starting but also stops existing NRPC users from opening another DB, and they also see the error message: "Server Error: Access to the server is restricted due to maximum number of users." Default is 0 (unlimited access to server by user). Recommendation: DEFAULT For further details, see "Server_MaxUsers" on page 217.

Domino system parameters	Definition
Server_Pool_Tasks	<p>Sets the number of threads assigned to each thread pool. Enables a small pool of threads to do server work instead of one per user.</p> <p>Default: The number of threads is 40 per port.</p> <p>In testing, 40 threads were able to handle 3000 clients with high efficiency. If your server handles fewer clients concurrently, you can improve performance by decreasing the number of threads in the pool proportionately.</p> <p>If you are using the server just as an HTTP server, drop the number of threads in the core server threadpool to a low value (for example, 10). In R5, the Domino HTTP server manages its own pool of threads, so any threads existing in the main server threadpool simply use up resources on the system, even if they are idle. The next release will be a little different.</p> <p>Default is 40</p> <p>Recommendation: DEFAULT</p>
Server_Session_Timeout=<minutes>	<p>Cause Domino to terminate an NRPC connection or session that has been idle for this amount of time. A session is considered idle if no activity detected. If it is set too low, it may cause the server to have to re-open database server sessions too often. The best setting for this parameter is determined by the server load, numbers of concurrent users on the server, etc.</p> <p>Beware, creating a new session is very CPU expensive; for example, re-using sessions is <i>much</i> more efficient, which is why we have the parameter.</p> <p>The term "Session" also includes server "internal" sessions, such as agents, replicator tasks, server to server and so on, so plan for it!</p> <p>Default: 4 hours (240)</p> <p>Recommendation: 30/45 minutes (but never less)</p> <p>For further details, see "Server_Session_Timeout" on page 218.</p>
Server_Show_Performance	<p>This parameter writes server performance events into the Notes Log when set =0</p> <p>Using the console command SHOW PERFORMANCE sets this parameter to =1 and the "Server performance monitoring is now enabled" message appears on the Domino server console. The statistics will run once a minute onto the console. To stop server performance events from being displayed on the console, type SHOW PERFORMANCE again and performance events will be toggled off and no longer be displayed on the console.</p> <p>On the Server Console the concurrent transactions will display as:</p> <p>03:41:55 PM 41 Transactions/Minute, 5 Users</p> <p>The FT function is quite similar to the WRKACTJOB command on the iSeries server.</p> <p>Default: <not applicable></p>
Server-Tasks=<task1>,<task2>,<task3>	<p>List of all ServerTasks Tasks that will start automatically at server startup. In addition to the default tasks Replica, Router, Update, Stats, clrepl, clbdbir, cladmin, you can run almost any other server task such as Compact, Event, and Fixup. Just be aware that each task increases the server's load.</p> <p>Note: This parameter is not automatically updated on install or upgrades and new releases.</p> <p>The tasks ICM, HTTP, IIOP, IMAP, POP3, LDAP, QNNINADD, DECS, LDAP, SMTPMTA, and NNTP are all configurable at the time a server is installed.</p> <p>For details, see Chapter 8, "Understanding the Domino server jobs" on page 229.</p>
skip_fixup=1	<p>5.0.4 Client is the only method of stopping auto-fixup removing all the docs from a damaged database and allowing some form of recovery, etc.</p> <p>Will be ignored in the server notes.ini.</p>

Domino system parameters	Definition
Translog_AutoFixup	<p>Only if Domino cannot use the TxL logs to fully recover a database will the FIXUP -J task be run to allow that DB to be accessed again. 0=DISABLED 1=ENABLED (Default)</p> <p>Notes:</p> <ul style="list-style-type: none"> ▶ Translog parameters are always overwritten by the values from the server document, so there is no point to changing them in the notes.ini directly. ▶ If FIXUP -J has to run after a crash, it assigns a new DBIID to that database, and notifies the administrator that a new full DB backup is required for that database + any others it has fixed. ▶ With this parameter <i>enabled</i>, this happens automatically during startup after a crash but only if the TxL recovery failed for some reason. ▶ When this parameter is <i>disabled</i>, Domino does not run the FIXUP -J task automatically. It just notifies the administrator to run the FIXUP task with the -J parameter on the corrupt database/s later!) and those databases will <i>not</i> be accessible. ▶ This parameter should only be <i>disabled</i> during debugging to allow you to take a copy of a corrupt DB before running FIXUP -J b hand. <p>Default=1 ENABLED Recommendation: ENABLED</p>
Translog_MaxSize= <size in Mb>	<p>Defines the maximum size in Mb that the collection of 64 Mb Circular TxL log files can grow to before being forced to start over-writing (re-using) themselves. This parameter plus Translog_UseAll are both ignored when using Archival TxL.</p> <p>Attention: The minimum size is 192 MB (it ignores all lower values!), so if you start Domino with physically less disk space for TxL logs than 192 Mb, it identifies a fatal error (panics)!</p> <p>TxL logs are written in 64k chunks. The logs “segment” (next log file “extent”) at 64 Mb. The logs are binary and created empty 64 Mb long. When re-used, they are incrementally renamed and the contents cleared down. Never routinely delete TxL log files! The only purpose of TxL on iSeries is fast recovery after a crash and incremental backups.</p> <p>Recommendation: Circular: 4096 (4Gb) (This is also the max value) Archival: 8192 (8Gb) Default: (3 x Physical RAM) For further details, see “Translog_MaxSize” on page 218.</p>
Translog_Path= <Full-Directory-Path>	<p>Directory path to location of TxL Log files. Default location is \logdir in data directory. However, on non-iSeries servers, we strongly recommend that you store the transaction log on a separate mirrored device, such as a RAID level 0 or 1 device with a dedicated controller.</p> <p>On iSeries, the Translog_Path is for convenience of locating your log files. It does not impact performance.</p> <p>Attention: Ensure TxL has sufficient rights to either create the directory named, or rights to use that directory. If TxL has insufficient rights, it considers that it has lost its log files and freezes rather than starting Fixup and trashing your DBs.</p> <p>Default=DATA\LOGDIR</p>

Domino system parameters	Definition
Translog_Performance	<p>Specifies the trade-off between transactional log runtime and restart recovery time, as follows:</p> <p>1 = Favor runtime. Checkpoint every 500 Mb. Flush after 500 Mb.</p> <p>2 = Standard (default) Checkpoint every 16 Mb, Flush at 49 Mb.</p> <p>3 = Favor restart recovery time. Checkpoint every 5 Mb, Flush at 15 Mb.</p> <p>Using a 500 Mb cache makes no sense with less than 500 Mb of logging space available. Default: 2 = (Standard) For further details, see "Controlling performance: Favor Run Time or Restart Recover Time" on page 377.</p>
TransLog_ReCreate_LogCTRL	<p>Used when the all the TxL logs (*.txn) exist, but the control file (logctrl.lfh) is missing. 1=ENABLED</p>
Translog_Status	<p>Translog_Status Enable TxL logging for all Release 5 databases 0 - Transactional logging disabled 1 - Transactional logging enabled You must upgrade databases to the Domino Release 5 format before they can be transactionally logged. Setting Translog_Status=0 disables transaction logging when the server is restarted, but the second restart re-enables it. Note: TxL can improve server overall performance in most cases, except on iSeries, where memory management and I/O are fully optimized anyway. TxL, however, dramatically improves restart times after a Domino crash (all platforms), improves data integrity, significantly reduces data loss after a crash, and allows online incremental and full backups using the TxL logs via a new Domino API for backup software developers. Default is 0 (DISABLED) Recommendation: ENABLED</p>
Translog_Style	<p>Type of transaction logging. 0 = Circular (default). The system continuously re-uses the extent log files, overwriting old TxLs 1 = Archive. The system does not immediately re-use extent log files until after you use a Domino R5 API backup utility to archive TxL logs. Recommendation: Either. Depends on needs. Note: Translog parameters are always overwritten by the values from the server document, so there is no point to changing them in the notes.ini directly.</p>

Domino system parameters	Definition
Translog_UseAll	<p>Allow TxL Logger to use all the available space to be found on the "Translog_Path=<directory> parameter" for Circular Transaction log files. 0=DISABLED (Default) 1=ENABLED</p> <p>Notes:</p> <ul style="list-style-type: none"> ▶ Translog parameters are always overwritten by the values from the server document, so there is no point to changing them in the notes.ini directly. ▶ This parameter and Translog_MaxSize are both ignored when using Archival TxL Logging, which keeps filling up your disk until either you run a backup or the disk runs out of space (resulting in Domino crashing). ▶ This parameter is also partially ignored even in Circular TxL because the maximum size for Circular is 4 GB, which is all it will ever grow to if you let it. ▶ On iSeries, you should always have this parameter <i>disabled</i> and you should always set the Translog_MaxSize parameter if using any form of TxL either Circular or Archival TxL logging. <p>Default=DISABLED Recommendation: DISABLED</p>
View_Rebuild_Dir=<path to desired directory>	<p>Changes the temporary work directory where the UPDATE task keeps all its work files and causes all views including (after R5.0.3) permuted (categorized) views to be rebuilt using the faster R5 Optimized view rebuild which uses the system TEMP directory (or the specified directory) to generate all temporary files (.DTF files) instead of the R4 default of the DATA directory. Some operating systems require a trailing directory separator. Convert all DBs to R5 to take full advantage.</p> <p>Recommendation: This directory be placed outside the data directory path preferably on its own ASP. For iSeries, also make sure qnotes owns the directory, and has full authority to it. Default: R4 standard rebuild. For details, see "View_Rebuild_Dir" on page 222.</p>

Table C-2 shows all the notes.ini logging parameters that are useful for debugging but have a considerable performance cost and should be left switched *off* when not required.

Table C-2 Logging parameters

Logging variable	Description
Log <parameter>	<p>Modifies logging behavior Variable 1 = Log's file name Variable 2 = Logging options to be added together 1= Log to console 2= Force database fixup when opening log 4= Full document scan (as opposed to quick scan or open) Variable 3 = Not currently used Variable 4 = Number of days at which to delete log documents Variable 5 = Size of log text in event documents</p>
Log_Replication	<p>Indicates level of replication session logging 0 - Do not log replication events 1 - Log that a database is replicating 2 - Log summary information about each database 3 - Log information about each replicated document (both design and data documents) 4 - Log information about each replicated field</p>

Logging variable	Description
Log_Sessions	Enable or Disable logging of all sessions as they start + stop; use for troubleshooting only. 0 - Do not log 1 - Log Sessions as they begin and end
Log_Tasks	Whether to log server task info to the console and log file. 0 - Do not send status information 1 - Send status of server tasks to log file and to the console
Log_Update	Level of index logging. 0 - Records when indexer starts and shuts down 1 - Records when the Indexer starts and shuts down + when Indexer updates views + full text indexes for specific databases. 2 - As 1 + record names of views the Indexer is updating.
Log_View_Events	Logging level for view rebuilds. 0 - Do not log messages when views are rebuilt 1 - Log messages when views are rebuilt
Mail_Log_To_MiscEvents	Determines if all mail event messages are displayed in the Miscellaneous Events view of the log file: 0 - Does not display mail events in Misc Events 1 - Displays mail events in the Miscellaneous Events view of log file
No_Force_Activity_Logging	Sets the database activity logging mode. Without this, set the Statlog server task enables the Record Activity option for every database on the server and adds 64 KB to each database. 0 - Allows automatic activity logging on all databases 1 - Prevents automatic activity logging on all databases
Passthru_LogLevel	Specifies the level of trace information recorded for all network connections (including passthru) in the Miscellaneous Events view of the log file. 0 - No information is recorded 1 - Only errors are recorded 2 - Summary progress information is recorded 3 - Detailed progress information is recorded 4 - Full trace information is recorded
PhoneLog	Settings to define which, if any, phone calls are written to the Notes Log. 0 = Don't log phone calls 1 = Log all phone calls except fails due to "busy" 2 = Log all phone calls
Platform_Statistics_Enabled=1	Track performance metrics of operating system + output results to server and statrep. Type show stat platform to display them. Disabled by default in R5.0.3 and later versions. PLATFORM <argument> The PLATFORM command controls this feature at the console using the following arguments: <ul style="list-style-type: none"> ▶ TIME <n> Change sample rate n (minutes) def 1 ▶ RESET Begin new stats session; reset metrics ▶ WAIT Pauses the collection of performance data ▶ RUN Resume the collection of performance data
RTR_Logging	Enables or disables monitoring of Cluster Replicator activity. 0 - Disables monitoring of the Cluster Replicator 1 - Enables monitoring of the Cluster Replicator
Show_Task_Detail	Enables additional information on the Show Task command.

Logging variable	Description
Client_Clock=1 Debug_Outfile= <path\filename> Debug_Console=1	Unsupported Client only parameter. Enable client NRPC calls, timings and sizing For details, see “Lotus Client NRPC Monitoring Tool Client_Clock” on page 157.
Console_Loglevel=	Logging level for the Domino server console. No information displayed 1 - Only errors are displayed 2 - Summary progress information is displayed 3 - Detailed progress information is displayed 4 - Full trace information is displayed
Log_AgentManager=	0 - Do not show logging 1 - To show partial and complete successes 2 - To show complete successes The Log_AgentManager setting provides you with a subset of the debugging information that Debug_AMgr generates. This option provides less output, but it has a smaller impact on performance. Some people keep the Log_AgentManager setting turned on even when there are no problems, just to have additional information in the log. If you have both notes.ini variables specified, Debug_AMgr settings will take precedence.
Log_DirCat=	Non-Zero=enabled (values 0 or 1) Allows you to log the dir cat task more frequently. Without this setting, the log file only shows when the DirCat starts.
Log_Disable_Session_Info=	Non-Zero=enabled (valid 1 or 0) Allows administrators to turn off the writing of session events to the log.
Log_MailRouting=	Degree of logging for the router task 0 - Note: Defaults to 20 10 - Shows errors, warning, and major events 20 - As 10 but successful deliveries + transfers also logged 30 - Add thread information 40 - Add transfer messages, message queues, + full document information for mail.boxes

Table C-3 lists the key notes.ini parameters related to agents that affect performance. For more information, refer to Julie Kadashevich’s frequently asked questions (FAQ) on everything you wanted to know about agents on Notes.net on the Web at:

<http://www.notes.net/46dom.nsf/bc218894c8b6c287852568560055be79/574c99ccb345839185256976004e811e?OpenDocument>

Table C-3 Agent Manager and the Adminp parameters that affect performance

Agent Manager variable	Function
AdminPInterval	Interval cycle (in minutes) for when the Administration Process (ADMINP) carries out many types of requests. This setting corresponds to the interval setting in the Administration Process section of a Server document. The Administration Process looks in the Server document first, and if it does not find a value in the Interval field, it refers to the notes.ini setting. Default: If setting omitted and no value is specified in the Server document, the interval cycle is 60 minutes.

Agent Manager variable	Function
AMgr_DocUpdateAgentMinInterval	<p>Specifies the minimum elapsed time (in minutes) between execution of the same document update-triggered agent.</p> <p>A delay between the execution of the same event-triggered agent (triggered by a document update or by new mail) exists so the user has control over the interval between the given agents. If the agent is triggered and its corresponding minimum interval hasn't passed, the Agent Manager schedules the task to run at its last run time plus the minimum interval.</p> <p>Default value is 30 minutes.</p>
AMgr_DocUpdateEventDelay	<p>Specify a delay of execution (in minutes) that the Agent Manager includes in the schedule for a document update-triggered agent after a document update event.</p> <p>This delay between when an event occurs (a document is updated or new mail is delivered) and the time at which the agent is scheduled to be executed, exists so that any rapid series of events will only result in one execution of a given agent. Then, the Agent Manager as well as the overall server can perform more efficiently.</p> <p>The default value is 5 minutes.</p>
AMgr_NewMailAgentMinInterval	<p>Specifies the minimum elapsed time (in minutes) between execution of the same new mail-triggered agent.</p> <p>A delay between the execution of the same event-triggered agent (triggered by a document update or by new mail) exists so the user has control over the interval between the given agents. If the agent is triggered and its corresponding minimum interval hasn't passed, the Agent Manager schedules the task to run at its last run time plus the minimum interval.</p> <p>The default value is 0.</p>
AMgr_NewMailEventDelay	<p>Specifies a delay of execution (in minutes) that the Agent Manager includes in the schedule for a new mail-triggered agent after a new mail message is delivered.</p> <p>This delay between when an event occurs (a document is updated or new mail is delivered) and the time at which the agent is scheduled to be executed, exists so that any rapid series of events will only result in one execution of a given agent. Then, the Agent Manager and the overall server can perform more efficiently.</p> <p>The default value is 1 minute.</p>
AMgr_SchedulingInterval	<p>Controls how quickly an agent gets into the schedule queue by specifying a delay (in minutes) between running of the Agent Manager's scheduler. Valid values: 1 minute to 60 minutes. The default value is 1 minute.</p> <p>The Agent Manager automatically uses the default values for this variable. To override the default values, you can add it to the server's notes.ini file. In general, this rarely require tuning.</p>
AMgr_UntriggeredMailInterval	<p>Controls how quickly an agent gets into the schedule queue by specifying a delay (in minutes) between running of the Agent Manager's check for untriggered mail. Incoming Mail (or indeed any notes document) is classed as "untriggered" when it's added to a database through replication, which does not cause a "new mail" trigger to fire. This parameter helps guarantee that the replicated mail will be processed by an agent.</p> <p>Valid values: 1 minute to 1440 minutes (the number of minutes in a day). The default value is 60 minutes.</p> <p>The Agent Manager automatically uses the default values for this variable. To override the default values, you can add it to the server's notes.ini file. In general, this rarely require tuning.</p>
DominoAsynchronizeAgents	<p>Specifies whether agents on a Web Server triggered by browser clients can run at the same time. Options are:</p> <ul style="list-style-type: none"> 0 - Only run one agent at a time (serially) 1 - Run more than one at a time (asynchronously) <p>Applies to: Browser clients</p>

Table C-4 shows all the Domino notes.ini parameters related to Indexing and full text (FT) indexing, all of which can and will have a significant effect on Domino performance.

Table C-4 Indexing and full text (FT) indexing parameters

Parameter	Definition
Default_Index_Lifetime_Days	Allows administrators to set a default lifetime for indexes associated with database views if none was selected by the database designer in the Design View Attributes dialog box ("Discard index on server after n days of inactivity"). Examples: Default_Index_Lifetime_Days=30 Valid on workstations Default: 45 days
DominoAnalyzeFormulas	Web pages that contain @functions can be cached through the Formula Analyzer, which "analyzes" the formula on the page to determine if it propagates changes frequently, and under what conditions the cached command becomes invalid. If the command propagates changes frequently or is prone to becoming invalid, then it will not be cached. Values: 0 or 1. Default=1 (ENABLED)
FT_Alternate_Filter	R5.0.3: Alternate filter added for attachment indexing which enhances the capabilities of the Keyview filter. This alternate filter provides support for file types, such as the Ichitaro file format, which is important in international regions. For example: FT_ALTERNATE_FILTER=na1tf1tr.dll
FT_Domain_Directory_Name	R5.03 Select location & name of Domain Index. Default is FTDOMAIN.DI. in Data Directory By adding this setting, Domino will support directory links and Index relocation.
FT_DOMAIN_IDXTHDS=	Specifies the number of indexing threads to use for Domain Search. Using more threads lets the Domain Catalog server index more files simultaneously, but requires more CPU utilization, and response to search queries may be slow. With fewer indexing threads, search speeds up because of greater CPU availability, but changes are not reflected in the index as quickly. Default: None, although if this setting is omitted, the default number of threads used is two per CPU. For example, a server with two CPUs uses four indexing threads by default when indexing. Do not exceed eight threads per server or you may degrade the performance of the server, even on servers with more than four CPUs.
FT_HTML_Title	By default, when Domino indexes HTML files for Domain Search, only the text in the <BODY> field is indexed. Domino does not index text included in any other fields. To include all HTML fields in a Domain index, for example, <META> or <TITLE> fields: Edit the FORMATS.INI file. Find the 210=htm statement & remark out the statement: REM 210=htm
FT_Index_Accent_Sens	Enables accent sensitive full text searching for special language characters. Default=0 (DISABLED)
FT_Index_Attachments	Disable indexing of attachments. Setting = 2 overrides any setting in a Notes database or the domain indexer and overrides any setting in a notes database or the domain indexer and turns off attachment indexing.
FT_Index_Ignore_Attachment_Types	R5.0.4: Control which type of attachments are not to be included in your Domain Index. A neat option, because files like .TIF and .JPG can consume considerable amounts of space. Add: =.exe, .tif, .jpg comma separated, restart server
FT_Intl_Setting	Imposes several limitations on full text functionality to let Notes work properly with the Japanese language. When enabled (set to 1), this setting turns off stemming, makes all full text indexes case-sensitive, and ignores the setting for the stop word file. Applies to workstations.
FT_KV_LOG=1	Switch this on to log output from a run to see if and why Keyview is failing to filter.

Parameter	Definition
FT_LIBName	<p>R5.0.5: Domino for iSeries now includes an alternate GTR search engine beginning in Release 5.0.5. This engine is not turned on by default. If you activate the new engine, you may see improvements in index creation and search response time, particularly with large databases or databases containing a mixture of different languages. However, a limitation is that the maximum document size that can be indexed will be reduced to approximately 5.3 MB, versus a maximum of about 8 MB with the original search engine. Documents that are too large will be excluded from the database's index.</p> <p>The GTR 34 engine is activated by placing the following parameter in the server's notes.ini file.</p> <p>FT_LIBName=ftgtr34</p> <p>Note: If you activate the alternate search engine, using this .ini parameter, Domino will re-build the indices for all databases that are currently indexed. If the parameter is later removed from the notes.ini file, all indices will again be rebuilt in their original R5 format. Default=<not applicable> See 7.6, "GTR search engine version 3.4" on page 199.</p>
FT_Max_Instances=<max # of words>	Governs the maximum unique keys found by a wildcard search. Default was 1000 now 15000
FT_Max_Instances_Large=	<p>R5: Control the maximum number of instances which can be indexed. Both must be taken into consideration: Notes uses this second one FT_MAX_INSTANCES_LARGE=< > if there is plenty of memory available. Otherwise, it will use FT_MAX_INSTANCES=< >. Current defaults are:</p> <ul style="list-style-type: none"> ▶ FT_MAX_INSTANCES 100000 ▶ FT_MAX_INSTANCES_LARGE 200000 <p>Both variables must be set. It is not sufficient to set one of the variables and not the other variable.</p>
FT_Max_Search_Results	<p>Increases the default return limit for FT searches Example: FT_MAX_SEARCH_RESULTS=10000 Default=5000</p>
FT_No_CompWinTitle	<p>Setting the variable to 1 will disable computing of the window title. This variable was introduced because computing a window title can be expensive in terms of CPU resources. By default, Domain Indexing will Compute Window titles. Many Notes Documents do not populate a Title, Subject, or Topic field. Consequently, the Domain Indexer assigns the title "Document has no title" to these documents. Options to avoid this are using Site Search, and to create a better title for the document by running the Note through the default View formula. Even if set, it is still possible to get a "Document has no title" warning. Default=0 (ENABLED)</p>
FT_Summ_Default_Language=english	<p>Set the Domain Search Summarizer language. Whenever the Summarizer assigns a language to a document, and is then unable to summarize in that language, it will then try to summarize using the default language instead. Whenever the Summarizer cannot summarize a document because it is too short, or it cannot summarize a document because it cannot summarize in the language it is trying to summarize in, no error message and no summary will be produced. Values: Any language, for example Dutch, English, French Valid: R5.0.3 NT Default=English (if null string) or if not present use Native language from locale</p>

Parameter	Definition
FT_Use_AltFiltr	<p>Enables the indexing of file attachments in the Ichitaro format. Full Text Search supports for attachments in the Ichitaro file format. The user must also set two values in the ftaf.ncf file: aftaro6 and aftaro9. (The ftaf.ncf file is in your Notes program directory. It is not automatically installed with Notes; the file will not exist unless the user or administrator has created it.)</p> <p>Note: ftaf.ncf file should be owned by QNOTES.</p> <p>Valid: R5.0.8 (Client + Server) on iSeries</p> <p>Default=0 DISABLED</p>
FTG_Index_Limit= X	<p>You can increase this index size limit by setting the value of X calculated as follows: 8MG is calculated as $8 * 1024 * 1024 = 8388608$; thus FTG_INDEX_LIMIT=8388608</p> <p>12MB is calculated as $12 * 1024 * 1024 = 12582912$; thus FTG_INDEX_LIMIT=12582912</p> <p>NOTE: There is no upper limit to the size.</p>
FTG_No_Summary	<p>Disable the Summarizer. Domain Search can, under certain circumstances, return a result to a user who cannot access the result document. If summarization is also enabled, this problem is exacerbated, because the result of a search will, if detailed results are specified, now include several key sentences from the document. If you have such a situation, you may want to disable the Summarizer by setting this variable to 1.</p> <p>Valid: R5.0.3 on NT</p> <p>Default=0 (ENABLED)</p>
FTV_Fields_<database>	<p>FTV_Fields_<database> specifies the numeric and date fields in a specific database that are full text indexed. Using this parameter, you want the ability to specify a path to a database located in a subdirectory.</p> <p>Not valid in R5</p>
FTV_Max_Fields	<p>Limiting Date and Number Fields to be Full Text Indexed</p> <p>Not valid in R5</p>
FullTextMultiProcess	<p>Allows multiple full-text Indexer server tasks to run on the server simultaneously. The <i>value</i> argument specifies the maximum number of Indexers that can run at the same time.</p> <p>Default=1</p>
QueryMaxResults	<p>Defines the maximum number of documents that the query engine will return. The query engine cannot handle a query that results in more than 5000 matches in 800 documents, and will return the error "Query is not understandable".</p>
Update_No_Fulltext	<p>Disables the full-text indexer, set to 1</p> <p>Default: If omitted, full-text indexing is ENABLED.</p> <p>Note: If you disable full-text indexing, you must restart the server for this change to take effect.</p>
Update_Suppression_Time =<minutes>	<p>Defines the delay time in minutes between full text index and view updates, even if immediate indexing is set.</p> <p>The server builds a queue of changed documents for the indexer and the agent manager, which wake up periodically and process from the queue. This <i>will</i> increase your CPU load significantly.</p> <p>Default=5 minutes</p>
Update_Suppression_Limit =<value>	<p>Overrides the Update_Suppression_Time variable if <value> number of duplicate requests are received to update indexes and views.</p> <p>The server builds a queue of changed documents for the indexer and the agent manager, which wake up periodically and process from the queue. This <i>will</i> increase your CPU load significantly.</p> <p>Default=<not applicable, uses suppress time></p>

Parameter	Definition
Updaters	<p>During startup, this fires up the heavy interactive-like UPDATE task on the iSeries side. By default, one job is started. The number of additional tasks is controlled with this parameter. Set this value according to your organization's requirements. The formula for calculating the number of tasks is:</p> <p><Number of CPUs in the box> - < 1></p> <p>Change this parameter only after careful planning. Multiples of these can have a serious impact on performance.</p>

Table C-5 shows the Mail related notes.ini parameters that will affect performance

Table C-5 The mail related notes.ini parameters that most affect performance

Parameter	Definition
MailClusterFailover	<p>This has the affect of forcing a server to try to deliver mail to the primary server and deliver it to the backup server if the primary is unavailable. The users will obviously have switched servers in line with their CLUSTER.NCF file when the primary server is unavailable.</p> <p>The delivery failover to the backup server does not happen until a number of connection failures have occurred on any server (that has the MailClusterFailover=1 option set in its notes.ini) trying to connect to the primary server. This is expected as the cost of routing has to increase for failover to occur. These failover times will also vary depending on whether the connecting server is in the same Notes Named Network.</p> <p>MailClusterFailover=1 allows failover (0 Disables)</p>
MailCompactDisabled	<p>Enables or disables the routine compacting of the server's mail.box. Without this setting in the notes.ini file, mail.box is compacted routinely when the COMPACT server task runs.</p> <p>0 - Enables compacting of mail.box 1 - Disables compacting of mail.box Default: Not applicable</p> <p>When disabled, the transfer/delivery threads are no longer shutdown and restarted. Previous behavior was that the router would shutdown the transfer/delivery threads and immediately restart them, simply skipping the compaction of mail.box.</p>
MailDisablePriority	<p>Used to disable high priory mail delivery (when requested by the sender) and treats all mail as if it were normal priority mail. 1=Disable. Default=0 (Allow High Priority Mail)</p>
MailLowPriorityTime= <time1-time2>	<p>Routes low-priority mail during the specified time. Without this setting in the notes.ini file, the Notes server routes low-priority mail only between 12 a.m. (midnight) and 6 a.m. Set the time range using 24-hour format. You must specify a time range for this setting; Notes won't deliver low-priority mail if you specify only one specific time, such as 4 p.m. For example: MailLowPriorityTime=21:00-23:00 Specifies that low-priority mail is routed from 9 p.m. to 11 p.m. Default=00:00-06:00</p>
MailMaxConcurrentXfer Threads	<p>This setting determines the maximum number of concurrent mail transfer threads per destination. The default is the MailMaxThreads value (see below) divided by 2. Note that in Domino Release 5, this setting is no longer included in notes.ini; it is a field in the Configuration Settings document.</p>
MailMaxDeliveryThreads	<p>Setting determines the maximum number of threads the Router can create to perform local mail delivery. The default is 1. Increasing this value can improve message throughput for local deliveries. The ideal number ranges from 3 to 25. This is determined by a formula, based upon the NSF_Buffer_Pool_Size</p> <p>Note: In Domino Release 5, this setting is no longer included in notes.ini; it is a field in the Configuration Settings document.</p>

Parameter	Definition
MailMaxThreads	Controls the number of concurrent outbound connections the Domino router task supports over each active port driver. If this parameter is omitted, Domino uses one thread per server port. Increasing this number creates more threads to handle mail transfers. However, additional threads may increase the demand for server processing time. In Domino Release 5, this setting is no longer included in notes.ini; it is a field in the Configuration Settings document. Default is one thread per server port. See also "MAILMAXTHREADS" on page 225.
Mail_Transfer_Retry_Minutes	This value applies to the "Route at once if X messages are pending" field on the Connection Record. It is designed to control the Retry Algorithm of the Mail transfer queue. When a remote server is unavailable or goes down in the middle of a session, the Retry algorithm kicks in and attempts to re-establish the connection with the remote server and complete the transfer. This parameter controls the number of minutes used before an attempt is made to retry a failed connection. Should a Scheduled Event fail the notes.ini MaxRetryDelay=X is used to control the total amount of time Notes will attempt to retry. The usefulness of this override is limited because the value is not an absolute. It can be overridden by other factors.
MaxRetryDelay	Used to control the total amount of time Notes will attempt to retry should a Scheduled Event fail. The usefulness of this override is limited because the value is not an absolute. It can be overridden by other factors. Default
Mail_Number_of_Mail_Boxes	Setting determines the number of mail files a user can have on the server. Note: In Domino Release 5, this setting is no longer included in notes.ini; it is a field in the Configuration Settings document.
MinNewMailPoll	Setting determines how often workstations can contact the server to see if new mail has arrived for the user. This setting overrides the user's selection in the Mail Setup dialog box. You can increase the mail polling interval if there are a large number of mail users on your server and you want to prevent frequent polling from affecting server performance.
NoMsgCache	Setting disables per-user message caching by the IMAP task. This can improve capacity (number of users) on a server by reducing memory consumption. However, response time for some user operations may be slower.
POP3_Config_Update_Interval	Setting determines how often (per minute) the Domino server that runs the POP3 service update its configuration information. Default= 2 minutes.
SecureMail	Setting this variable to 1 causes the Notes mailer to sign and encrypt all mail sent from the workstation and removes the Sign and Encrypt options from all dialog boxes.
SharedMail	Specifies whether the shared mail "feature" is used for new mail delivered to this server: 0 - Shared mail not used for new mail 1 - Shared mail used for new mail to this server. 2 - Shared mail for new mail to this server and for new mail transferred through this server Default: 0 (shared mail not used) Recommendation: Avoid if at all possible.

Table C-6 shows all the clustering parameters that will affect performance.

Table C-6 Clustering parameters

Parameter	Definition
Server_TransInfo_Max	The number of intervals to average the transaction counts. The availability of cluster servers and the process of load-based failover is based on the comparison of the Server_Availability_Index and Server_Availability_Threshold. The Server_Availability_Index is calculated by the following formula: $100 - (\text{transactions} / \text{Server_Transinfo_Normalize})$ Transactions are counted within the statistic interval Server_Transinfo_Update_Interval. The number of intervals that are used to average the transaction counts may be set by Server_TransInfo_Max. Default is 5 Recommendation: No change
Server_TransInfo_Update_Interval	Transactions counted within the statistic interval. The availability of cluster servers and the process of load-based failover is based on the comparison of the Server_Availability_Index and Server_Availability_Threshold. The Server_Availability_Index is calculated by the following formula: $100 - (\text{transactions} / \text{Server_Transinfo_Normalize})$ Transactions are counted within the statistic interval Server_Transinfo_Update_Interval. The number of intervals that are used to average the transaction counts may be set by Server_TransInfo_Max. Default is 15 seconds Recommendation: No change
Server_Cluster_Probe_Timeout	Default 1 (minute) max 120: The interval in which the cluster servers probe each other for availability. Can cause serious performance issues if set too fast depending on loading.
CLREPL_Poll_Interval	Cluster Polling Interval (Seconds) Attention: The default and very large values in some earlier revisions are reset to a very small value that can cause continuous polling and clustering. Tip: Manually set this to a sensible value.
Cluster_Replicators= Disable_Cluster_Replicator=1	RNext: parameter to defines the number of Cluster Replicators and another to enable/disable clustering. (1=Enable 0=Disable) The R5 Cluster Administrator is now a server thread and performs and replaces all the former CLADMIN functions (CLREPL & CLDBDIR) each time the server starts. Note: CLREPL and CLDBDIR will be removed from notes.ini ServerTasks on upgrade to R5. Excessive or unnecessary cluster replication will seriously impact performance.
RTR_Cached_Handle_Disable	Enables or disables the caching of open databases in a cluster. 0 - Enables caching of open databases 1 - Disables caching of open databases Cached handles improves response times, but can cause excessive I/O on very busy systems.
rtr_initial_retry_interval rtr_max_retry_interval	Cluster initial retry and max allowed interval. The default was a very large values in some earlier revs and gets reset to very small value causing initial value to override the maximum retry interval. Tip: Manually set this to a sensible value.

Paremeter	Definition
Server_Availability_Threshold=<%>	<p>Works directly with the Domino statistic Server.Availability. When Server.Availability reaches the threshold set in the Server_Availability_Threshold parameter, the server begins rejecting user requests. The threshold may need to be set high (95 to 97) for the best failover characteristics.</p> <p>In addition, Server_Transinfo_Normalize may need to be set. This allows you to "normalize" the response times experienced by your Domino server and insures that the failover processing will occur only when it's supposed to. This value can be tailored for your environment.</p> <p>Tip: Manually set to a carefully calculated value.</p>
Server_transinfo_Normalize	<p>Used when calculating the server availability index to "normalize" the response times observed at the server (that is, it divides the observed response times by this normalize value). Until now, this setting was undocumented, but it is available in both R4.6 and R5.</p> <p>For the availability index calculation to work properly, the normalize value should be roughly equal to the average Domino transaction time (for the server in question) in milliseconds*100. The default value is 3000ms, corresponding to an average response time of 30ms per transaction.</p> <p>Note: This default setting was appropriate for "the average server" when clustering was first shipped several years ago, but it is too large for the current generation of servers. You should use a lower normalize value with today's faster servers, so loads failover correctly.</p> <p>Tip: Manually set to a sensible value.</p>
Server_Cluster_Default_Port =TCPIP	<p>This directs cluster communications between the servers through TCP/IP. Even if your network uses multiple protocols, you should configure all servers in a single cluster to support the same set of network protocols. This ensures that clients can connect to any server in the cluster in the event of failover or for workload balancing. You should specify the TCP/IP port of the server on the Server_Cluster_Default_Port setting in the notes.ini file of all servers that belong to a cluster.</p>
Server_Cluster_On=<0 or 1>	<p>Enables clustering when set to 1. We recommend that you use TCP/IP for cluster server configurations for optimum communications between the servers in the cluster. Although the failover and workload balancing logic was designed and implemented to be independent of the actual network protocol, Lotus only fully certifies Domino clusters running with TCP/IP.</p>
Server_Cluster_Probe_Port=<port-name>	<p>Determines the communications port the cluster manager uses for availability probes. The release notes for some versions of Domino instruct you to set the Server_Cluster_Default_Port and this one. This is an error in the documentation; Domino does not use the Server_Cluster_Probe_Port setting. Doesn't actually do anything in current revisions, but you can put it in if you choose.</p>



Domino R5 statistics

Table D-1 shows the entire R5 statistics table and their meaning. These names and some of the following descriptions can be located on any Domino R5 server in the “names and messages (advanced)” / “statistic names” view inside events4.nsf.

Table D-1 Domino R5 statistics and their descriptions

R5 statistic name	Statistic functional description
Agent.Daily.AccessDenials	Number of daily access denials for agents
Agent.Daily.ScheduledRuns	Number of daily scheduled agent runs
Agent.Daily.TriggeredRuns	Number of daily triggered agent runs
Agent.Daily.UnsuccessfulRuns	Number of daily unsuccessful agent runs
Agent.Daily.UsedRunTime	Number of seconds to run agent
Agent.Hourly.AccessDenials	Number of hourly access denials for agents
Agent.Hourly.ScheduledRuns	Number of hourly scheduled agent runs
Agent.Hourly.TriggeredRuns	Number of hourly triggered agent runs
Agent.Hourly.UnsuccessfulRuns	Number of hourly unsuccessful agent runs
Agent.Hourly.UsedRunTime	Number of seconds to run agent
Calendar.Total.All.Appointments.Reservations	Total appointments and reservations in scheduling
Calendar.Total.All. Users.Resources	Total users and resources in scheduling
Calendar.Total.Appts	Total appointments in scheduling
Calendar.Total.Reservations	Total reservations in scheduling
Calendar.Total.Resources	Number of resources in scheduling
Calendar.Total.Users	Total users in scheduling
Collector.Time.Collectd	Time statistics were collected
Collector.Time.Elapsed	Time since collector was started

R5 statistic name	Statistic functional description
Comm.NumOldSessionsClosed	Number of sessions closed by the Notes server to accommodate new users
Database.BufferControlPool.Peak	Peak number of database control pools used
Database.BufferControlPool.Size	Size of database buffer control pool
Database.BufferControlPool.Used	Number of database control pools used
Database.BufferPool.Allocated	Number of database control pools allocated
Database.BufferPool.Maximum	Maximum size of database control pools
Database.BufferPool.Peak	Peak number of database buffer pools
Database.BufferPool. PerCentReadsInBuffer	Percentage of buffer pool reads
Database.BufferPool.Reads	Number of database buffer pool reads
Database.BufferPool.Used	Number of used database buffer pool
Database.BufferPool.Writes	Number of database buffer pool writes
DataBase.DbCache.CurrentEntries	Number of entries in the database cache at this time
DataBase.DbCache.HighWaterMark	High water mark of the database cache
DataBase.DbCache.Hits	Number of hits to the database cache
DataBase.DbCache.InitialDbOpens	Number of Db opens done by the database cache
DataBase.DbCache.Lookups	Number of lookups to the database cache
DataBase.DbCache.MaxEntries	Max number of entries in the database cache
DataBase.DbCache.OvercrowdingRejections	Number of rejections due to the overcrowding of the database
Database.ExtMgrPool.Peak	Peak number of external manager pools
Database.ExtMgrPool.Used	Number of External Manager pools
Database.NIFPool.Size	Size of database NIF pool
Database.NIFPool.Used	Number of database NIF pools
Database.NSF.FreeHandleStack.FreeHandleStackHits	Number of free handle stack hits
Database.NSF.FreeHandleStack.HandleAllocations	Number of database free handle stack allocations
Database.NSF.FreeHandleStack.MissRate	Free handle stack miss rate
Database.NSFPool.Peak	Peak number of database NSF pools
Database.NSFPool.Size	Size of database NSF pool
Database.NSFPool.Used	Number of database NSF pools
Database.RM.Current.K.Restart.Redo.Hold	Changes sitting on log but not in database(s) yet
Database.RM.Current.K.UnCommitted.Undo.Hold	Changes held back by long running transaction(s)
Database.RM.Current.MaxChkptScan.msec	Slowest time to create a checkpoint
Database.RM.LastChkpt.Avg.K.Logged.PerTran	The size of transactions

R5 statistic name	Statistic functional description
Database.RM.LastChkpt.Interval.Sec	Time since last checkpoint (Ex. 27 hours. This is an Idle machine)
Database.RM.LastChkpt.K.Logged	Amount logged in Checkpoint interval (usually near Database.RM.Sys.M.Chkpt.Interval)
Database.RM.LastChkpt.Time	"07/07/1999 09:31:10" PDT Time of checkpoint
Database.RM.LastChkpt.Total.DBs	Databases open or in DBCache at time of checkpoint
Database.RM.LastChkpt.Trans	Actual transactional logging type transactions (API calls that did DB updates)
Database.RM.Peak.Avg.K.Logged.PerTran	The size of transactions
Database.RM.Peak.Interval.Sec	Time since last checkpoint
Database.RM.Peak.K.Logged	Amount logged in Checkpoint interval (usually near Database.RM.Sys.M.Chkpt.Interval)
Database.RM.Peak.Time	"06/01/1999 08:14:47" PDT Time of checkpoint
Database.RM.Peak.Total.DBs	Databases open or in DBCache at time of checkpoint
Database.RM.Peak.Trans	Actual transactional logging type transactions (API calls that did database updates)
Database.RM.Peak.Trans.PerMin	Transactions/Min
Database.RM.Restart.Duration.Sec	Time to apply the incomplete changes after a crash (in seconds)
Database.RM.Restart.K.Applied	Amount of incomplete changes detected and applied in KB (Ex. 22MB in 33 sec.)
Database.RM.Restart.Time	"06/01/1999 08:11:59" PDT Time the server was started
Database.RM.SinceChkpt.K.Logged	Logged records since last checkpoint (Kb)
Database.RM.SinceChkpt.Trans	Transactions since last checkpoint
Database.RM.SinceStartup.Aborts	Failed transactions (usually caused by some corrupt database or Notes in database. Look at log.nsf Misc Events if this is a large value)
Database.RM.SinceStartup.Critical.Log.Times	Shows how many times your system ran critically log full. If non-zero, consider increasing log size
Database.RM.SinceStartup.Log.Recs.Analyzed	(internal)
Database.RM.SinceStartup.Log.Recs.Applied	Log records applied (Redo=>Undo)
Database.RM.SinceStartup.Log.Recs.Read	Log record has been read
Database.RM.SinceStartup.Log.Recs.Written	Log record has been written
Database.RM.SinceStartup.M.Logged	How much log has been written in MB
Database.RM.SinceStartup.Trans	Actual transactional logging type transactions (API calls that did database updates)
Database.RM.Sys.Log.Type	Circular Logging style (Circular or Archive)
Database.RM.Sys.Logged	Enabled If Disabled, no other Database.RM statistics are shown

R5 statistic name	Statistic functional description
Database.RM.Sys.M.Chkpt.Interval	Checkpoint interval
Database.RM.Sys.M.Log.Size	Max Log size defined
Database.RM.Sys.M.Redo.Limit	Recommended max Redo.Hold span before aggressive flushing occurs
Database.RM.Sys.Phase	Normal would show if Media Recover was under way
Disk.C.Free	Free space on drive C:
Disk.C.Size	Size of disk C
Disk.D.Free	Free space on drive D:
Disk.D.Size	Size of disk D
Disk.Swapdisk.Free	Number of free bytes on the disk that contains the server's swap file
Domino.BuildName	Build name for this Domino server
Domino.BuildNumber	Build number for this Domino server
Domino.BuildPlatform	OS platform for this Domino server
Domino.BuildVersion	Build version for this Domino server
Domino.Command.CreateDocument	Count of "OpenDocument" URLs that have come into the server
Domino.Command.DeleteDocument	Count of "DeleteDocument" URLs that have come into the server
Domino.Command.EditDocument	Count of "EditDocument" URLs that have come into the server
Domino.Command.Login	Count of "Login" URLs that have come into the server
Domino.Command.Navigate	Count of "Navigate" URLs that have come into the server
Domino.Command.OpenAbout	Count of "OpenAbout" URLs that have come into the server
Domino.Command.OpenAgent	Count of "OpenAgent" URLs that have come into the server since server was
Domino.Command.OpenDatabase	Count of "OpenDatabase" URLs that have come into the server
Domino.Command.OpenDocument	Count of "OpenDocument" URLs that have come into the server
Domino.Command.OpenElement	Count of "OpenElement" URLs that have come into the server
Domino.Command.OpenForm	Count of "OpenForm" URLs that have come into the server
Domino.Command.OpenHelp	Count of "OpenHelp: URLs that have come into the server
Domino.Command.OpenIcon	Count of "OpenIcon" URLs that have come into the server

R5 statistic name	Statistic functional description
Domino.Command.OpenNavigator	Count of “OpenNavigator” URLs that have come into the server
Domino.Command.OpenServer	Count of “OpenServer” URLs that have come into the server
Domino.Command.OpenView	Count of “OpenView” URLs that have come into the server
Domino.Command.ReadForm	Count of “ReadForm” URLs that have come into the server
Domino.Command.SearchSite	Count of “SearchSite” URLs that have come into the server
Domino.Command.Total	Count of all URLs that have come into the server
Domino.Command.Unknown	Count of unknown URLs that have come into the server
Domino.Requests.Per1Day.Peak	Peak requests over the past day
Domino.Requests.Per1Day.PeakTime	Time of peak requests over the past day
Domino.Requests.Per1Day.Total	Total requests over the past day
Domino.Requests.Per1Hour	Total requests over the past hour
Domino.Requests.Per1Hour.Peak	Peak requests over the past hour
Domino.Requests.Per1Hour.PeakTime	Time of peak requests over the past hour
Domino.Requests.Per1Hour.Total	Total requests over the past hour
Domino.Requests.Per1Minute	Total requests over the past minute
Domino.Requests.Per1Minute.Peak	Peak requests over the past minute
Domino.Requests.Per1Minute.PeakTime	Time of peak requests over the past minute
Domino.Requests.Per1Minute.Total	Total requests over the past minute
Domino.Requests.Per5Minute	Total requests over the past 5 minutes
Domino.Requests.Per5Minute.Peak	Peak requests over the past 5 minutes
Domino.Requests.Per5Minute.PeakTime	Time of peak requests over the past 5 minutes
Domino.Requests.Per5Minute.Total	Total requests over the past 5 minutes
Domino.Requests.Total	Total requests
ICM.Command.Redirects.ClusterBusy	Number of commands received when all servers are busy
ICM.Command.Redirects.Successful	Number of successful redirects
ICM.Command.Redirects.Unsuccessful	Number of unsuccessful redirects
ICM.Command.Total	Total number of commands received
ICM.Command.Unknown	Total number of unknown commands received
ICM.Receive.Error	Total number of commands not received due to error
ICM.Requests.Per1Hour.Total	Number of requests received in the last hour

R5 statistic name	Statistic functional description
ICM.Requests.Per1Minute.Total	Number of requests in the last one minute
ICM.Requests.Per5Minutes.Total	Number of requests in the last 5 minutes
ICM.Sessions.Threads.Busy	Number of ICM server tasks currently running
ICM.Sessions.Threads.Idle	Number of ICM server tasks currently idle
ICM.Sessions.Total	Total number ICM server tasks since server startup
IMAP.Sessions.Accept.Queue	IMAP listener work queue of incoming sessions to be processed
IMAP.Sessions.Active	Current number IMAP server tasks
IMAP.Sessions.Inbound.BytesReceived	Total Number of BytesReceived by this IMAP server
IMAP.Sessions.Inbound.BytesSent	Total Number of BytesSent by this IMAP server
IMAP.Sessions.Inbound.Non-SSL	Number of IMAP server inbound TCP connections (non SSL)
IMAP.Sessions.Inbound.SSL	Number of IMAP Server inbound SSL connections
IMAP.Sessions.Inbound.SSL.Bad_Handshake	Number of IMAP server SSL Handshake Failures
IMAP.Sessions.Inbound.Total	Total Number of IMAP server inbound connections (SSL and non SLL)
IMAP.Sessions.Peak	Peak number IMAP server tasks since server startup
IMAP.Sessions.Threads.Busy	Number of IMAP server tasks currently running
IMAP.Sessions.Threads.Idle	Number of IMAP server tasks currently idle
IMAP.Sessions.Total	Total number IMAP server tasks since server startup
LDAP.Anonymous LDAP Connections	Anonymous connection count
LDAP.Average LDAP Search time	Average of all successful search times
LDAP.Failed LDAP Connections	Failed connections count
LDAP.Longest LDAP Search request	Search value for longest search
LDAP.Longest LDAP Search time	Longest individual search time
LDAP.Sessions.Accept.Queue	LDAP Listener work queue of incoming sessions to be processed
LDAP.Sessions.Active	Current number LDAP server tasks
LDAP.Sessions.Inbound.BytesReceived	Total Number of BytesReceived by this LDAP server
LDAP.Sessions.Inbound.BytesSent	Total Number of BytesSent by this LDAP server
LDAP.Sessions.Inbound.Non-SSL	Number of LDAP server inbound TCP connections (non SSL)
LDAP.Sessions.Inbound.SSL	Number of LDAP server inbound SSL connections
LDAP.Sessions.Inbound.SSL.Bad_Handshake	Number of LDAP server SSL Handshake failures
LDAP.Sessions.Inbound.Total	Total Number of LDAP server inbound connections (SSL and non SSL)

R5 statistic name	Statistic functional description
LDAP.Sessions.Peak	Peak number LDAP server tasks since server startup
LDAP.Sessions.Threads.Busy	Number of LDAP server tasks currently running
LDAP.Sessions.Threads.Idle	Number of LDAP server tasks currently idle
LDAP.Sessions.Total	Total number LDAP server tasks since server startup
LDAP.Simple LDAP Connections	Simple connection counter
LDAP.Strong Authentication Connections	Strong authentication connection count
LDAP.Total LDAP Abandons	Count of abandoned operations
LDAP.Total LDAP Adds	Count of attempted add operations
LDAP.Total LDAP Compares	Count of attempted compare operations
LDAP.Total LDAP Connections	Total of all connections
LDAP.Total LDAP Deletes	Count of attempted delete operations
LDAP.Total LDAP Extended Operations	Count of attempted extended operations
LDAP.Total LDAP Modifies	Count of attempted modify operations
LDAP.Total LDAP ModifyDNs	Count of attempted modifydn operations
LDAP.Total LDAP Searches	Count of attempted searched performed
LDAP.Total LDAP Searches for Root DSE	Count of search requests for root DSE
LDAP.Total LDAP Searches for Subschema	Count of search requests for subschema entry
Mail.AverageDeliverTime	Average time for mail delivery.
Mail.AverageServerHops	Average number of server hops for mail delivery.
Mail.AverageSizeDelivered	Average size of mail delivered.
Mail.Dead	Number of dead (undeliverable) messages in mail.box
Mail.Delivered	Number of mail messages moved into mail.box by router
Mail.Deliveries	Count of actual mail items delivered (may be different from delivered)
Mail.Domain	Mail domain name
Mail.Hold	Number of mail messages in message queue on hold
Mail.IMAP.Cache.MsgInsertAttempts	Number of IMAP server attempts to insert message in cache
Mail.IMAP.FindCacheAttempted	Number of attempts by IMAP server to find message in cache
Mail.IMAP.NumCacheHits	Number of messages IMAP server found in cache
Mail.IMAP.NumMsgInsertedInCache	Number of messages IMAP server inserted into cache
Mail.MaximumDeliverTime	Maximum time for mail delivery
Mail.MaximumServerHops	Maximum number of server hops for mail delivery
Mail.MaximumSizeDelivered	Maximum size of mail delivered

R5 statistic name	Statistic functional description
Mail.MinimumDeliverTime	Minimum time for mail delivery
Mail.MinimumServerHops	Minimum number of server hops for mail delivery
Mail.MinimumSizeDelivered	Minimum size of mail delivered
Mail.PeakByteTransferRate	Peak transfer rate
Mail.PeakMessagesTransferred	Peak number of messages transferred
Mail.PeakMessageTransferRate	Peak message transfer rate
Mail.PeakMessageTransferRate.Time	Time of peak message transfer rate
Mail.PeakTotalBytesTransferred	Highest number of bytes transferred
MAIL.TotalFailures	Total number of mail failures
Mail.TotalKBTransferred	Total mail transferred in KB
Mail.TotalPending	Number of mail messages pending
Mail.TotalRouted	Number of mail messages moved from mail.box to other servers
Mail.TotalRouted.SMTP	Number of mail messages moved from mail.box via SMTP
Mail.TotalRouter.NRPC	Number of mail messages moved from mail.box via NRPC
Mail.TransferFailures	Number of mail messages router was unable to transfer
Mail.Transferred	Number of mail messages router attempted to transfer
Mail.Waiting	Number of outgoing mail messages currently in mail.box waiting
Mail.WaitingforDNS	Number of mail messages in mail.box waiting for DNS
Mail.WaitingRecipients	Number of pending mail messages in mail.box awaiting delivery into user's
MailByDest.<destServerName>.Delivered	Total number of delivered messages
MailByDest.<destServerName>.TotalFailures	Total number of mail failures
MailByDest.<destServerName>.TotalRouted	Total number of mail failures
Mem.Allocated	Total amount of memory allocated by the server
Mem.Allocated.Process	Total amount of non-shared memory allocated by individual processes
Mem.Allocated.Shared	Total amount of server memory allocated as shared memory
Mem.Availability	Availability of server memory
Mem.Free	Amount of free memory available to the server
Mem.FTVDKMaxBufferSizeUsed	FTVDK Max buffer size used
Mem.PhysicalRam	Total amount of physical memory on the server
Mem.Quota	Maximum memory that can be allocated by the server

R5 statistic name	Statistic functional description
Mem.SwapFile.Size	Size (in bytes) of system swap file
Mem.Timeouts	Memory timeouts
Monitor.<EventName>.Failure	Event queue failure
Monitor.<EventName>.Fatal	Event queue fatal error
Monitor.<EventName>.Warning(High)	Event queue failure warning
Monitor.<Taskname>.Heartbeat	Event heartbeat count
MTA.ccMail.Dead	Dead messages in ccMail MTA Inbound and Outbound work queues
MTA.ccMail.TotalKBTransferred	Total number of Kilobytes (Inbound and Outbound)
MTA.ccMail.TotalRouted	Total recipients in ccMail MTA messages routed (Inbound and Outbound)
MTA.ccMail.TransferFailures	Total ccMail MTA transmission failures resulting in an NDR or UMN
MTA.ccMail.Transferred	Total ccMail MTA messages transferred (Inbound and Outbound)
MTA.ccMail.Waiting	Total messages waiting in ccMail MTA work queues
MTA.ccMail.WaitingRecipients	Number of Responsible Recipients in waiting mail held in Inbound and Outbound work queues
MTA.Smtp.Dead	Dead messages in SMTP/MIME MTA Inbound and Outbound work queues
MTA.Smtp.TotalKBTransferred	Total number of Kilobytes (Inbound and Outbound)
MTA.Smtp.TotalRouted	Total recipients in SMTP/MIME MTA messages routed (Inbound and Outbound)
MTA.Smtp.TransferFailures	Total SMTP/MIME MTA transmission failures resulting in an NDR or UMN
MTA.Smtp.Transferred	Total SMTP/MIME MTA messages transferred (Inbound and Outbound)
MTA.Smtp.Waiting	Total messages waiting in SMTP/MIME MTA work queues
MTA.Smtp.WaitingRecipients	Number of Responsible Recipients in waiting mail held in Inbound and Outbound work queues
NET.<PortName>.BytesReceived	Number of network bytes received
NET.<PortName>.BytesSent	Number of network bytes sent
NET.<PortName>.Sessions.Established.Incoming	Number of Incoming sessions established
NET.<PortName>.Sessions.Established.Outgoing	Number of Outgoing sessions established
NET.<PortName>.Sessions.Limit	Number of sessions at limit
NET.<PortName>.Sessions.LimitMax	Number of sessions at maximum limit
NET.<PortName>.Sessions.LimitMin	Number of sessions at minimum limit

R5 statistic name	Statistic functional description
NET.<PortName>.Sessions.Peak	Peak number of sessions
NET.<PortName>.Sessions.Recycled	Number of sessions that have been recycled
NET.<PortName>.Sessions.Recycling	Number of sessions recycling
NET.Log.<username>.UnwrittenEntries	Number of unwritten log notes waiting to be written
NET.TCP.BytesReceived	Number of network bytes received
NET.TCP.BytesSent	Number of network bytes sent
NET.TCP.Sessions.Established.Incoming	Number of Incoming sessions established
NET.TCP.Sessions.Established.Outgoing	Number of Outgoing sessions established
NET.TCP.Sessions.Limit	Number of sessions at limit
NET.TCP.Sessions.LimitMax	Number of sessions at maximum limit
NET.TCP.Sessions.LimitMin	Number of sessions at minimum limit
NET.TCP.Sessions.Peak	Peak number of sessions
NET.TCP.Sessions.Recycled	Number of sessions that have been recycled
NET.TCP.Sessions.Recycling	Number of sessions recycling
NET.TCPIP.BytesReceived	Number of network bytes received
NET.TCPIP.BytesSent	Number of network bytes sent
NET.TCPIP.Sessions.Established.Incoming	Number of Incoming sessions established
NET.TCPIP.Sessions.Established.Outgoing	Number of Outgoing sessions established
NET.TCPIP.Sessions.Limit	Number of sessions at limit
NET.TCPIP.Sessions.LimitMax	Number of sessions at maximum limit
NET.TCPIP.Sessions.LimitMin	Number of sessions at minimum limit
NET.TCPIP.Sessions.Peak	Peak number of sessions
NET.TCPIP.Sessions.Recycled	Number of sessions that have been recycled
NET.TCPIP.Sessions.Recycling	Number of sessions recycling
NET.VINES.BytesReceived	Number of network bytes received
NET.VINES.BytesSent	Number of network bytes sent
NNTP.<RemoteServerName>.Articles.Posted	Number of news articles posted by the server "RemoteServerName"
NNTP.<RemoteServerName>.Articles.Sent	Number of news articles sent to the server "RemoteServerName"
NNTP.<RemoteServerName>.Bytes.Received	Number of bytes received from the server "RemoteServerName"
NNTP.<RemoteServerName>.Bytes.Sent	Number of bytes sent to the server "RemoteServerName" by the NNTP
NNTP.<RemoteServerName>.Pull.Articles.Failed	Number of failed news article transfers from the server "RemoteServerName"

R5 statistic name	Statistic functional description
NNTP.<RemoteServerName>.Pull.Articles.Offered	Number of news articles offered by the server "RemoteServerName"
NNTP.<RemoteServerName>.Pull.Articles.Requested	Number of news articles requested from server "RemoteServerName"
NNTP.<RemoteServerName>.Pull.Articles.Transferred	Number of news articles transferred from "RemoteServerName" to the NNTP
NNTP.<RemoteServerName>.Push.Articles.Failed	Number of failed news article transfers to server "RemoteServerName"
NNTP.<RemoteServerName>.Push.Articles.Offered	Number of news articles offered to the server "RemoteServerName"
NNTP.<RemoteServerName>.Push.Articles.Requested	Number of news articles requested by the server "RemoteServerName"
NNTP.<RemoteServerName>.Push.Articles.Transferred	Number of news articles transferred to the server RemoteServerName"
NNTP.Articles.Posted	Number of news articles posted to the NNTP server
NNTP.Articles.Sent	Number of news articles sent from the NNTP server
NNTP.Bytes.Received	Number of bytes received by NNTP server
NNTP.Bytes.Sent	Number of bytes sent from NNTP server
NNTP.Pull.Articles.Failed	Number of failed news article transfers received by the NNTP server
NNTP.Pull.Articles.Offered	Number of news articles offered by the NNTP server during pull feeds
NNTP.Pull.Articles.Requested	Number of news articles requested from the NNTP server during pull feeds
NNTP.Pull.Articles.Transferred	Number of news articles transferred by the NNTP server during pull feeds
NNTP.Push.Articles.Failed	Number of failed news article transfers by the NNTP server during push
NNTP.Push.Articles.Offered	Number of news articles offered by the NNTP server during push feeds
NNTP.Push.Articles.Requested	Number of news articles requested from the NNTP server during push feeds
NNTP.Push.Articles.Transferred	Number of news articles transferred by the NNTP server during push feeds
NNTP.Sessions.Accept.Queue	NNTP listener work queue of incoming sessions to be processed
NNTP.Sessions.Active	Current number NNTP server tasks
NNTP.Sessions.Inbound.BytesReceived	Total Number of BytesReceived by this NNTP server
NNTP.Sessions.Inbound.BytesSent	Total Number of BytesSent by this NNTP server
NNTP.Sessions.Inbound.Non-SSL	Number of NNTP server inbound TCP connections (non SSL)

R5 statistic name	Statistic functional description
NNTP.Sessions.Inbound.SSL	Number of NNTP server inbound SSL connections
NNTP.Sessions.Inbound.SSL.Bad_Handshake	Number of NNTP server SSL Handshake Failures
NNTP.Sessions.Inbound.Total	Total Number of NNTP server inbound connections (SSL and non-SSL)
NNTP.Sessions.Outgoing.non-SSL	Total Outgoing non-SSL TCP Connections
NNTP.Sessions.Outgoing.SSL	Total Outgoing SSL TCP Connections
NNTP.Sessions.Peak	Peak number NNTP server tasks since server startup
NNTP.Sessions.Threads.Busy	Number of NNTP server tasks currently running
NNTP.Sessions.Threads.Idle	Number of NNTP server tasks currently idle
NNTP.Sessions.Total	Total number NNTP server tasks since server startup
Object.Mailobj.NSF.SharedBy.01	Number of objects shared by one user
Object.Mailobj.NSF.SharedBy.xx	Number of objects shared by xx users
Object.Mailobj.NSF.SharedBy.20	Number of objects shared by 20 users
Object.Mailobj.NSF.SharedBy.Greater	Number of objects shared by more than 20 users
Platform.LogicalDisk._Total.1._Total.1.PctTime	The percent time that all configured disks in all ASPs are busy. The first "_Total" refers to all physical disks, and the second "_Total" refers to all logical disks. For the iSeries, read it as the total of all disks configured.
Platform.LogicalDisk._Total.1._Total.1.PctTime.avg	The average of the percent time that all configured disks in all ASPs are busy. The first "_Total" refers to all physical disks, and the second "_Total" refers to all logical disks. For the iSeries, read it as the total of all disks configured.
Platform.LogicalDisk._Total.1._Total.1.PctTime.max	The max of the percent time that all configured disks in all ASPs are busy. The first "_Total" refers to all physical disks, and the second "_Total" refers to all logical disks. For the iSeries, read it as the total of all disks configured.
Platform.LogicalDisk._Total.1._Total.1.PctTime.min	The min of the percent time that all configured disks in all ASPs are busy. The first "_Total" refers to all physical disks, and the second "_Total" refers to all logical disks. For the iSeries, read it as the total of all disks configured.
Platform.LogicalDisk._Total.1._Total.1.PctUsed	The percentage of all configured disks in all ASPs that is currently allocated for data on iSeries
Platform.LogicalDisk._Total.1.AvgQueueLength	The average number of both read and write requests that were queued for all logical disks on all physical disks during the sample interval
Platform.LogicalDisk._Total.1.AvgQueueLength.avg	The average of the average number of both read and write requests that were queued for all logical disks on all physical disks during the sample interval
Platform.LogicalDisk._Total.1.AvgQueueLength.max	The maximum of the average number of both read and write requests that were queued for all logical disks on all physical disks during the sample interval

R5 statistic name	Statistic functional description
Platform.LogicalDisk._Total.1.AvgQueueLength.min	The minimum of the average number of both read and write requests that were queued for all logical disks on all physical disks during the sample interval
Platform.LogicalDisk._Total.1.PctTime	The percentage of the sampling interval that all logical disks on all physical disks are servicing read or write requests
Platform.LogicalDisk._Total.1.PctTime.avg	The average percentage of the sampling interval that all logical disks on all physical disks are servicing read or write requests
Platform.LogicalDisk._Total.1.PctTime.max	The maximum percentage of the sampling interval that all logical disks on all physical disks are servicing read or write requests
Platform.LogicalDisk._Total.1.PctTime.min	The minimum percentage of the sampling interval that all logical disks on all physical disks are servicing read or write requests
Platform.Memory.FaultsPerSec	The number of page faults per second in the memory pool this iSeries server is using
Platform.Memory.FaultsPerSec.avg	The average number of page faults per second in the memory pool this iSeries server is using
Platform.Memory.FaultsPerSec.max	The maximum number of page faults per second in the memory pool this iSeries server is using
Platform.Memory.FaultsPerSec.min	The minimum number of page faults per second in the memory pool this iSeries server is using
Platform.Memory.KBFree	Currently Available KB of physical memory available ("free") to processes
Platform.Memory.KBFree.avg	Average amount in KB of physical memory available ("free") to processes
Platform.Memory.KBFree.max	Maximum KB amount of physical memory ever available to ("free") processes
Platform.Memory.KBFree.min	Minimum KB is the lowest amount of physical memory ("free") that was available to processes
Platform.Memory.KBSize	The amount of memory in KB allocated to the iSeries server memory pool that this Domino partition is running in
Platform.Memory.PagesPerSec	The number of memory pages per second read from or written to disk
Platform.Memory.PagesPerSec.avg	The average number of memory pages per second read from or written to disk
Platform.Memory.PagesPerSec.max	The maximum (peak) number of memory pages per second read from or written to disk
Platform.Memory.PagesPerSec.min	The minimum (lowest) number of memory pages per second read from or written to disk

R5 statistic name	Statistic functional description
Platform.Memory.Wait->Ineligible	The number of Wait-to-Ineligible transitions for threads in the memory pool this iSeries server is using. Any value greater than zero indicates your max active in the memory pool is too low.
Platform.Memory.WaitToIneligible	The number of Wait-to-Ineligible transitions for threads in the memory pool this iSeries server is using. Any value greater than zero indicates your max active in the memory pool is too low.
Platform.PagingFile._Total.Util	Current page file utilization percentage
Platform.PagingFile._Total.Util.avg	Average page file utilization percentage
Platform.PagingFile._Total.Util.max	Max (peak) page file utilization percentage
Platform.PagingFile._Total.Util.min	Min (lowest) page file utilization percentage
Platform.Process.ADMINP.1.Util	The CPU utilization of the first adminp process of this iSeries server partition.
Platform.Process.AMGR.1.Util	The CPU utilization of the first agent manager process of this iSeries server partition
Platform.Process.AMGR.2.Util	The CPU utilization of the second agent manager process of this iSeries server partition
Platform.Process.AMGR.3.Util	The CPU utilization of the third agent manager process of this iSeries server partition
Platform.Process.AMGR.4.Util	The CPU utilization of the fourth agent manager process of this iSeries server partition
Platform.Process.HTTP.1.Util	The CPU utilization of the first http process of this iSeries server partition
Platform.Process.nadminp.1.Util	The CPU utilization of the first adminp process
Platform.Process.nadminp.2.Util	The CPU utilization of the second adminp process
Platform.Process.nadminp.3.Util	The CPU utilization of the third adminp process
Platform.Process.nadminp.4.Util	The CPU utilization of the fourth adminp process
Platform.Process.namgr.1.Util	The CPU utilization of the first amgr process
Platform.Process.namgr.2.Util	The CPU utilization of the second amgr process
Platform.Process.namgr.3.Util	The CPU utilization of the third amgr process
Platform.Process.namgr.4.Util	The CPU utilization of the fourth amgr process
Platform.Process.nhttp.1.Util	The CPU utilization of the first http process.
Platform.Process.nhttp.2.Util	The CPU utilization of the second http process.
Platform.Process.nhttp.3.Util	The CPU utilization of the third http process
Platform.Process.nhttp.4.Util	The CPU utilization of the fourth http process
Platform.Process.nreplica.1.Util	The CPU utilization of the first replica process
Platform.Process.nreplica.2.Util	The CPU utilization of the second replica process
Platform.Process.nreplica.3.Util	The CPU utilization of the third replica process

R5 statistic name	Statistic functional description
Platform.Process.nreplica.4.Util	The CPU utilization of the fourth replica process
Platform.Process.nrouter.1.Util	The CPU utilization of the first router process
Platform.Process.nrouter.2.Util	The CPU utilization of the second router process
Platform.Process.nrouter.3.Util	The CPU utilization of the third router process
Platform.Process.nrouter.4.Util	The CPU utilization of the fourth router process
Platform.Process.nserver.1.Util	The CPU utilization of the first server process. This is the percentage of the sample interval that the process used the CPU.
Platform.Process.REPLICA.1.Util	The CPU utilization of the first replica process of this iSeries server partition
Platform.Process.REPLICA.2.Util	The CPU utilization of the second replica process of this iSeries server partition
Platform.Process.REPLICA.3.Util	The CPU utilization of the third replica process of this iSeries server partition
Platform.Process.REPLICA.4.Util	The CPU utilization of the fourth replica process of this iSeries server partition
Platform.Process.ROUTER.1.Util	The CPU utilization of the first router process of this iSeries server partition
Platform.Process.SERVER.1.Util	CPU utilization of the first iSeries server process of this partition. This is the percentage of the sample interval that the process used the CPU.
Platform.Process.SERVER.2.Util	CPU utilization of the second iSeries server process of this partition. This is the percentage of the sample interval that the process used the CPU.
Platform.Process.SERVER.3.Util	CPU utilization of the third iSeries server process of this partition. This is the percentage of the sample interval that the process used the CPU.
Platform.Process.SERVER.4.Util	CPU utilization of the fourth iSeries server process of this partition. This is the percentage of the sample interval that the process used the CPU.
Platform.Process.UPDATE.1.Util	The CPU utilization of the first update process of this iSeries server partition
Platform.Process.UPDATE.2.Util	The CPU utilization of the second update process of this iSeries server partition
Platform.Process.UPDATE.3.Util	The CPU utilization of the third update process of this iSeries server partition
Platform.Process.UPDATE.4.Util	The CPU utilization of the fourth update process of this iSeries server partition
Platform.System.TotalUserUtil	% Total User Time is the average percentage of non-idle time all processors
Platform.System.TotalUserUtil.avg	% Total User Time is the average percentage of non-idle time all processors

R5 statistic name	Statistic functional description
Platform.System.TotalUserUtil.max	% Total User Time is the average percentage of non-idle time all processors
Platform.System.TotalUserUtil.min	% Total User Time is the average percentage of non-idle time all processors
Platform.System.TotalUtil	The current percent CPU utilization of all CPUs on the system from current total util sample
Platform.System.TotalUtil.avg	The average percent CPU utilization of all CPUs on the system. Average of all TotalUtil samplings for a session, calculated by summing all TotalUtil samplings, and dividing by number of samplings.
Platform.System.TotalUtil.max	The maximum percent CPU utilization of all CPUs on the system, which is the largest TotalUtil sampling for a session
Platform.System.TotalUtil.min	The minimum percent CPU utilization of all CPUs on the system, for example, the smallest TotalUtil sampling for a session
POP3.Command.<commandname>	POP3 command
POP3.Sessions.Accept.Queue	POP3 listener work queue of incoming sessions to be processed
POP3.Sessions.Active	Current number POP3 server tasks
POP3.Sessions.Inbound.BytesReceived	Total Number of BytesReceived by this POP3 server
POP3.Sessions.Inbound.BytesSent	Total Number of BytesSent by this POP3 server
POP3.Sessions.Inbound.Non-SSL	Number of POP3 server inbound TCP connections (non SSL)
POP3.Sessions.Inbound.SSL	Number of POP3 server inbound SSL connections
POP3.Sessions.Inbound.SSL.Bad_Handshake	Number of POP3 server SSL Handshake Failures
POP3.Sessions.Inbound.Total	Total Number of POP3 server inbound connections (SSL and non SSL)
POP3.Sessions.Peak	Peak number POP3 server tasks since server startup
POP3.Sessions.Threads.Busy	Number of POP3 server tasks currently running
POP3.Sessions.Threads.Idle	Number of POP3 server tasks currently idle
POP3.Sessions.Total	Total number POP3 server tasks since server startup
POP3.UserCache.Attempts	POP3 attempt to find user in cache
POP3.UserCache.Hits	Found in cache
POP3.UserCache.Inserts	Successful insert into cache
POP3.UserCache.Lookups	POP3 successful lookup user in cache
POP3.UserCache.OpenReasonCacheFull	Cache full
POP3.UserCache.OpenReasonDBChanged	Reason for open is database changed
POP3.UserCache.OpenReasonNewEntry	Reason for open is new entry

R5 statistic name	Statistic functional description
QOS.<TCP Service Type>.<Server Name>.<Monitor Id>.ResponseTime	TCP Probe response time
QOS.Mail.ISpy.<servername>.ResponseTime	Mail Probe response time
QOS.Mail.ISpy.ProbeError	Mail Probe Error
Replica.Cluster.CachedHandles	Number of handles cached by cluster
Replica.Cluster.Docs.Added	Number of docs added by Cluster Replicator
Replica.Cluster.Docs.Deleted	Number of docs deleted by Cluster Replicator
Replica.Cluster.Docs.Updated	Number of docs updated by Cluster Replicator
Replica.Cluster.Failed	Number of failed replications since server startup
Replica.Cluster.Files.Local	Number of cluster replicas on this server
Replica.Cluster.Files.Remote	Number of cluster replicas on other servers in cluster
Replica.Cluster.Retry.Skipped	Number of times the Cluster Replicator did not attempt replication because it
Replica.Cluster.Retry.Waiting	Number of replicas currently waiting for replication retry attempts
Replica.Cluster.SecondsOnQueue	Number of seconds Cluster Replicator was on queue
Replica.Cluster.SecondsOnQueue.Avg	Average number of seconds Cluster Replicator was on queue
Replica.Cluster.SecondsOnQueue.Max	Maximum number of seconds Cluster Replicator was on queue
Replica.Cluster.Servers	Number of other servers in the cluster that are receiving replications from this
Replica.Cluster.SessionBytes.In	Number of incoming bytes to the Cluster Replicator
Replica.Cluster.SessionBytes.Out	Number of outgoing bytes to the Cluster Replicator
Replica.Cluster.Successful	Number of successful replications since server startup
Replica.Cluster.WorkQueueDepth	Current number of modified databases in the queue waiting to be replicated
Replica.Cluster.WorkQueueDepth.Avg	Average number of modified databases in the queue waiting to be replicated
Replica.Cluster.WorkQueueDepth.Max	Maximum number of modified databases in the queue waiting to be replicated
Replica.Docs.Added	Number of documents added to the server's databases via replication.
Replica.Docs.Deleted	Number of documents deleted from the server's databases via replication
Replica.Docs.Updated	Number of documents updated in the server's databases via replication
Replica.Failed	Number of attempted replications that returned some kind of error

R5 statistic name	Statistic functional description
Replica.Successful	Number of replication attempts that returned no error of any kind
Reporter.Time.Analysis	Time statistics were analyzed
Reporter.Time.Collectected	Time statistics were collected
Reporter.Time.Elapsed	Time since reporter was started
SEM.Timeouts	OS Sem timeout
Server.<servernameinfo>	Server access error
Server.Administrator	Name of administrator (from server record in Name and Address Book
Server.AvailabilityIndex	Current percentage index of a server's availability. Value range is 0-100. Zero
Server.AvailabilityThreshold	Current setting of a server's availability threshold.
Server.BootID	Unique number identifying this boot session
Server.BusyTimeQuery.ReceivedCount	Number of queries to busytime in scheduling
Server.BusyTimeQuery.Returned.Objects.Schedules	Number of responses to queries to busytime in scheduling
Server.Cluster.AvailabilityIndex	Current percentage index of a server's availability. Value range is 0-100. Zero
Server.Cluster.AvailabilityThreshold	Current setting of a server's availability threshold
Server.Cluster.Name	Name of the cluster to which the server belongs
Server.Cluster.OpenRedirects.Failover.Successful	Number of times that a server successfully redirects a client to another cluster
Server.Cluster.OpenRedirects.Failover.Unsuccessful	Number of times that a server is not able to redirect a client to another cluster
Server.Cluster.OpenRedirects.FailoverByPath.Successful	Number of times a server successfully redirects a client to another cluster
Server.Cluster.OpenRedirects.FailoverByPath.Unsuccessful	Number of times a server is not able to redirect a client to another cluster
Server.Cluster.OpenRedirects.LoadBalance.Successful	Number of times that a server successfully redirects a client to another cluster
Server.Cluster.OpenRedirects.LoadBalance.Unsuccessful	Number of times that a server is unable to redirect a client to another cluster
Server.Cluster.OpenRedirects.LoadBalanceByPath.Successful	Number of times that a server successfully redirects a client to another cluster
Server.Cluster.OpenRedirects.LoadBalanceByPath.Unsuccessful	Number of times that a server is not able to redirect a client to another cluster
Server.Cluster.OpenRequest.ClusterBusy	Number of times that a client tries to open a database on the server
Server.Cluster.OpenRequest.DatabaseOutOfService	Number of times a client tries to open a database that is marked

R5 statistic name	Statistic functional description
Server.Cluster.OpenRequest.LoadBalanced	Number of times a client tries to open a database on the server
Server.Cluster.Portname	Name of the default port that is used for intracuster network traffic
Server.Cluster.ProbeCount	Number of times that a server completes a probe request of the other cluster
Server.Cluster.ProbeError	Number of times that a server receives an error when probing another server
Server.Cluster.ProbeTimeout(mins)	Interval at which the intracuster probe or cluster member heartbeat occurs
Server.Cluster.Trans.IntervalAvgTime	Average interval time of cluster transactions
Server.Cluster.Trans.IntervalInMinutes	Interval between cluster transactions
Server.Cluster.Trans.IntervalInSeconds	Interval between cluster transactions
Server.Cluster.Trans.IntervalsUseInAvg	Current setting for the number of intervals used to capture transaction data
Server.Cluster.Trans.LastIntervalAvgTime	Last average interval time of cluster transactions
Server.Cluster.Trans.NormalizeValue	Current setting of acceptable average transaction time in milliseconds
Server.Cluster.Trans.RunningAvgTime	Average total running time of cluster transactions
Server.Cluster.Trans.RunningCount	Number of cluster transactions
Server.Cluster.Trans.RunningTime	Total running time of cluster transactions
Server.Coprocessor	Yes or No, depending on whether the server contains a math coprocessor
Server.CPU.Count	Count of processors in system
Server.CPU.Type	Type of system processor (for example, 386, 486, Pentium)
Server.Description	Server's one line description from server's Name & Address book
Server.Location	Physical location of server
Server.MinIdleDisconnect	Minimum number of minutes for disconnect
Server.MinIdleDisconnect.Time	Time at which minimum number of minutes for disconnect happened.
Server.Monitor.Start	Time at which server monitor started
Server.Name	Server name
Server.OpenRequest.Maxusers	Open requests for that have reached maximum users
Server.OpenRequest.PreV4Clients	Open requests for pre-R4 clients
Server.OpenRequest.Restricted	Open requests for restricted clients
Server.OpenRequest.V4Clients	Open requests for R4 clients

R5 statistic name	Statistic functional description
Server.Path.Data	Server's default data drive and directory
Server.Path.Swap	Complete path of swap file
Server.Ports	Server ports
Server.Ports.RS232	Number of RS232 (serial) ports on the server
Server.PoweredBy.Notes	Self explanatory
Server.Sessions.Dropped	Sessions dropped in mid-transaction
Server.Task	This statistic appears once for every task running on the server
Server.Tasks	Number of tasks currently running on the server
Server.Time.Start	Time at which server was started
Server.Title	Server title
Server.Trans.PerMinute	Number of server transactions in the last minute
Server.Trans.PerMinute.Peak	Highest number of transactions to occur during a one-minute interval
Server.Trans.PerMinute.Peak.Time	Time that peak transactions per minute occurred
Server.Trans.Total	Number of transactions processed since server started
Server.User	This statistic occurs once for each user, listing the user name, session ID
Server.Users	Number of users with open sessions on the server
Server.Users.1MinPeak	Peak number of users in one minute
Server.Users.1MinPeakTime	Time when peak users in one minute was reached
Server.Users.5MinPeakPeak	Peak number of users in five minutes
Server.Users.5MinPeakTime	Time when peak users in five minutes was reached
Server.Users.Peak	Peak number of concurrent users since server was started
Server.Users.Peak.Time	Time that last peak users occurred
Server.Version.MTBF	Version of MTBF running on the server
Server.Version.NLM	Version of NetWare running on the server
Server.Version.Notes	Version of Lotus Notes running on the server
Server.Version.NT	Version of Windows NT running on the server
Server.Version.OS	Version of OS2 running on the server
Server.Version.OS2	Version of OS/2 running on the server
Server.WorkThreads	Maximum number of concurrent transactions to create
SMTP.<name>.Ave	SMTP average
SMTP.<name>.Count	SMTP Count

R5 statistic name	Statistic functional description
SMTP.<name>.Max	SMTP maximum
SMTP.<name>.Min	SMTP.Minimum
SMTP.<name>.Total	SMTP total
SMTP.Command.<CommandName>	SMTP command
SMTP.MessagesProcessed	Number of SMTP messages processed
SMTP.Sessions.Accept.Queue	SMTP listener work queue of incoming sessions to be processed
SMTP.Sessions.Active	Current number SMTP server tasks
SMTP.Sessions.Inbound.BytesReceived	Total Number of BytesReceived by this SMTP server
SMTP.Sessions.Inbound.BytesSent	Total Number of BytesSent by this SMTP server
SMTP.Sessions.Inbound.Non-SSL	Number of SMTP server inbound TCP connections (non SSL)
SMTP.Sessions.Inbound.SSL	Number of SMTP server inbound SSL connections
SMTP.Sessions.Inbound.SSL.Bad_Handshake	Number of SMTP server SSL Handshake Failures
SMTP.Sessions.Inbound.Total	Total Number of SMTP server inbound connections (SSL & non SSL)
SMTP.Sessions.Peak	Peak number SMTP server tasks since server startup
SMTP.Sessions.Threads.Busy	Number of SMTP server tasks currently running
SMTP.Sessions.Threads.Idle	Number of SMTP server tasks currently idle
SMTP.Sessions.Total	Total number SMTP server tasks since server startup
SMTPMTA.ConvFailures	Total SMTP MTA conversion failures
SMTPMTA.Dead	Dead SMTP MTA messages
SMTPMTA.Delivered	Total SMTP MTA messages delivered
SMTPMTA.HighInBound	Highest number of inbound processes
SMTPMTA.HighOutBound	Highest number of outbound processes
SMTPMTA.InBoundBytes	Total number of bytes inbound
SMTPMTA.InBoundSize	Largest message inbound
SMTPMTA.OutBoundBytes	Total number of bytes inbound
SMTPMTA.OutBoundSize	Largest message outbound
SMTPMTA.TotalRouted	Total SMTP MTA messages routed
SMTPMTA.TransFailures	Total SMTP MTA transmission failures
SMTPMTA.Transferred	Total SMTP MTA messages transferred
SMTPMTA.Waiting	Total SMTP MTA messages waiting
SMTPMTA.WaitingConv	Mail waiting to be converted
SMTPMTA.WaitingRecipients	Mail waiting to be delivered

R5 statistic name	Statistic functional description
SMTPMTA.WaitingTrans	Mail waiting to be transferred
SNTP.<name>.min	Stamp Network TadPole Minibus
SPIN.AvtlterToHit	Average of Spin Lock Iterations to Hits
SPIN.Delays	Count of Spin Lock Delays
SPIN.DelaysMsecs	SpinLock Delay in msecs
SPIN.FRWSemGetReadExt	SpinLock FRW semaphore read
SPIN.FRWSemPutReadExt	SpinLock FRW semaphore put
SPIN.Hits	Count of Spin Lock Hits
SPIN.MaxDelayMask	SpinLock max delay mask
SPIN.MaxDelayMsecs	SpinLock max delay in msecs
Stats.Time.Current	Time statistics package is updated
Stats.Time.Start	Time statistics package was started
Web.Retrieve.Access.FTP	Number of FTP retriever accesses
Web.Retrieve.Access.Gopher	Number of Gopher retriever accesses
Web.Retrieve.Access.HTTP	Number of HTTP retriever accesses
Web.Retrieve.Bytes.Received	Number of retriever bytes received
Web.Retrieve.Bytes.Sent	Number of retriever bytes sent
Web.Retrieve.ImageCache.Hits	Number of successful accesses on retriever image cache
Web.Retrieve.ImageCache.Misses	Number of unsuccessful accesses on retriever image cache
Web.Retrieve.LogLevel	Web retriever logging Level
Web.Retrieve.LogMessages	Web retriever logging messages? (Yes/No)
Web.Retrieve.Process.Access.HTTP	Number of retriever process HTTP accesses
Web.Retrieve.Process.Access.FTP	Number of retriever process FTP accesses
Web.Retrieve.Process.Access.Gopher	Number of retriever process Gopher accesses
Web.Retrieve.Process.Bytes.Received	Number of bytes received by retriever process
Web.Retrieve.Process.Bytes.Sent	Number of bytes sent by retriever process
Web.Retrieve.Process.Num.Active	Number of active retriever processes
Web.Retrieve.Process.Num.Busy	Number of busy retriever processes
Web.Retrieve.Process.Num.Idle	Number of idle retriever processes
Web.Retrieve.Process.Num.Maximum	Maximum number of retriever processes
Web.Retrieve.Process.ProcessID	Process ID number
Web.Retrieve.Process.State	State of retriever processes

R5 statistic name	Statistic functional description
Web.Retriever.Process.URLs.Failed	Number of failed retriever process URLs
Web.Retriever.Process.URLs.Requested	Number of requested retriever process URLs
Web.Retriever.Process.URLs.Succeeded	Number of successful retriever process URLs
Web.Retriever.Time.Current	Current time
Web.Retriever.Time.Duration	Elapsed time since retriever started
Web.Retriever.Time.Start	Time retriever started
Web.Retriever.URLs.Failed	Number of retriever URLs that have failed
Web.Retriever.URLs.Requested	Number of retriever URLs that have been requested
Web.Retriever.URLs.Succeeded	Number of retriever URLs that have been successfully accessed
Web.Retriever.Version	Web retriever version
Web.Retriever.VPOOL.Max.Buf	Number of buffer VPOOLS that have reached maximum size
Web.Retriever.VPOOL.Max.Element	Number of element VPOOLS that have reached maximum size
Web.Retriever.VPOOL.Max.Marker	Number of marker VPOOLS that have reached maximum size
Web.Retriever.VPOOL.Max.Text	Number of text VPOOLS that have reached maximum size
Web.Retriever.VPOOL.Max.URL	Number of URL VPOOLS that have reached maximum size

The following list shows the server commands followed by their abbreviation in parentheses (). For the complete syntax and parameters of these commands, refer to the *Lotus Domino Administrator's Guide* or the Domino 5 Administration Help database (help5_admin.nsf).

Broadcast (b)
 Dbcache Flush (db f)
 Dbcache Show (db s)
 Drop (dr)
 Exit (e)
 Help (he)
 Load (l)
 Pull (pul)
 Push (pus)
 Quit (q)
 Replicate (rep)
 Restart Port (res po)
 Restart Server (res se)
 Route (ro)
 Set Configuration (se c)
 Set Secure (se se)
 Set Statistics (se st)
 Show Allports (sh a)
 Show Cluster (sh cl)

Show Configuration (sh co)
Show Directory (sh dir)
Show Diskspace (sh dis)
Show Memory (sh me)
Show Performance (sh pe)
Show Port (sh po)
Show Schedule (sh sc)
Show Server (sh se)
Show Stat (sh st)
Show Tasks (sh ta)
Show Users (sh u)
Show Transactions (sh tr)
Start Port (sta po)
Stop Port (sto po)
Tell (t)
Trace (tr)

Related publications

The publications listed in this section are considered particularly suitable for a more detailed discussion of the topics covered in this redbook.

IBM Redbooks

For information on ordering these publications, see “How to get IBM Redbooks” on page 481.

All Domino redbooks covering performance related topics

- ▶ *Measuring Lotus Notes Response Times with Tivoli's ARM Agents*, SG24-4787
- ▶ *Image and Workflow Library: Capacity Planning and Performance Tuning for VisualInfo and Digital Library Servers*, SG24-4974
- ▶ *Performance Considerations for Domino Applications*, SG24-5602
- ▶ *Getting the Most From Your Domino Directory*, SG24-5986

iSeries server-related Redbooks

- ▶ *AS/400 Program Calls with Domino LotusScript: An Example*, REDP0004
- ▶ *Lotus Domino for AS/400 R4.6: Installation, Customization and Administration*, REDP0032
- ▶ *Lotus QuickPlace for AS/400: Setup and Management Considerations*, REDP0045
- ▶ *iNotes Access for Microsoft Outlook on Domino for iSeries: A Migration Guide*, REDP0126
- ▶ *Business-to-Business Integration Guide: Using WebSphere Application Server and Domino for iSeries*, REDP0139
- ▶ *Integrating WebSphere Commerce Suite with Domino Back-End Application: iSeries 400 Edition*, REDP0141
- ▶ *iSeries Handbook Updates: Changes from April 2001 to August 2001*, REDP0143
- ▶ *Backup Recovery and Media Services for OS/400: More Practical Information*, REDP0508
- ▶ *IBM @server iSeries Handbook Version 5 Release 1*, GA19-5486-21
- ▶ *AS/400 Server Capacity Planning*, SG24-2159
- ▶ *AS/400 Electronic-Mail Capabilities*, SG24-4703
- ▶ *AS/400 Performance Management*, SG24-4735
- ▶ *AS/400 Performance Explorer Tips and Techniques*, SG24-4781
- ▶ *Mail Integration for Lotus Domino 4.5 on the IBM Integrated PC Server for AS/400 V3R2 or V3R7*, SG24-4977
- ▶ *A Fast Path to AS/400 Client/Server Using AS/400 OLE DB Support*, SG24-5183
- ▶ *AS/400 Consolidation Strategies and Implementation*, SG24-5186
- ▶ *AS/400 IBM Network Station: Techniques for Deployment in a WAN*, SG24-5187
- ▶ *AS/400 Client Access Express for Windows: Implementing V4R4M0*, SG24-5191
- ▶ *Lotus Domino for AS/400: Integration with Enterprise Applications*, SG24-5345
- ▶ *How to Replace OfficeVision/400 in Your Applications: Looking at Domino for AS/400 and AS/400 Alternatives*, SG24-5406

- ▶ *Lotus Domino for AS/400 R5: Implementation*, SG24-5592
- ▶ *iSeries e-business Handbook: A Technology and Product Reference*, SG24-5694
- ▶ *Lotus Fax for Domino for AS/400: Getting the Straight Facts*, SG24-5941
- ▶ *Lotus Domino for AS/400 Internet Mail and More*, SG24-5990
- ▶ *Lotus Domino for AS/400: Problem Determination Guide*, SG24-6051
- ▶ *Developing an e-business Application Using Lotus Domino for AS/400*, SG24-6052
- ▶ *Application Service Provider Business Model: Implementation on the iSeries Server*, SG24-6053
- ▶ *AS/400e to IBM @server iSeries Migration: A Guide to System Upgrades at V4R5 and V5R1*, SG24-6055
- ▶ *Consolidating Windows 2000 Servers in iSeries: An Implementation Guide for the IBM Integrated xSeries Server for iSeries*, SG24-6056
- ▶ *AS/400 Remote Access Configuration Examples*, SG24-6058
- ▶ *New Enterprise Integration Functions for Lotus Domino for AS/400*, SG24-6203
- ▶ *Exchange Migration and iNotes Implementation on the iSeries Server*, SG24-6230
- ▶ *iSeries e-business Handbook V5R1 Technology and Product Reference*, SG24-6711

Lotus Domino and other Lotus software

- ▶ *Building a Portal with Lotus Domino R5*, REDP0019
- ▶ *Domino Certification Authority and SSL Certificates*, REDP0046
- ▶ *Notes and Domino Connectivity - A Collection of Examples*, REDP0115
- ▶ *International Standards - A Document Imaging Review*, GG24-2544
- ▶ *Exploring The IBM eNetwork Communications Suite*, SG24-2111
- ▶ *Enterprise Integration with Domino.Connect*, SG24-2181
- ▶ *Developing Web Applications Using Lotus Notes Designer for Domino 4.6*, SG24-2183
- ▶ *From Client/Server to Network Computing, A Migration to Java*, SG24-2247
- ▶ *Lotus Notes Release 4 In a Multiplatform Environment*, SG24-4649
- ▶ *Enterprise Calendaring with Lotus Notes: The Notes to OfficeVision Connection*, SG24-4811
- ▶ *Lotus Solutions for The Enterprise, Volume 1 Lotus Notes: An Enterprise Application Platform*, SG24-4837
- ▶ *The Domino Defense: Security in Lotus Notes 4.5 and the Internet*, SG24-4848
- ▶ *LotusScript for Visual Basic Programmers*, SG24-4856
- ▶ *Secrets to Running Lotus Notes: The Decisions No One Tells You How to Make*, SG24-4875
- ▶ *Lotus Notes Release 4.5: A Developer's Handbook*, SG24-4876
- ▶ *From Client/Server to Network Computing, A Migration to Domino*, SG24-5087
- ▶ *Java Thin-Client Programming for a Network Computing Environment*, SG24-5115
- ▶ *The Next Generation in Messaging: Moving from MS Exchange to Lotus Notes and Domino*, SG24-5167
- ▶ *Publishing Tools in the Network Computing Framework*, SG24-5205
- ▶ *Internet Security in the Network Computing Framework*, SG24-5220

- ▶ *IBM Web-to-Host Integration Solutions*, SG24-5237
- ▶ *Lotus Solutions for the Enterprise, Volume 5 NotesPump: The Enterprise Data Mover*, SG24-5255
- ▶ *VisualAge for Java Enterprise Version 2: Data Access Beans - Servlets - CICS Connector*, SG24-5265
- ▶ *Lotus Domino Release 5.0: A Developer's Handbook*, SG24-5331
- ▶ *Eight Steps to a Successful Messaging Migration: A Planning Guide for Migrating to Lotus Notes and Domino*, SG24-5335
- ▶ *Lotus Notes and Domino R5.0 Security Infrastructure Revealed*, SG24-5341
- ▶ *Using VisualAge for Java to Develop Domino Applications*, SG24-5424
- ▶ *Connecting Domino to the Enterprise Using Java*, SG24-5425
- ▶ *Lotus Domino R5.0 Enterprise Integration: Architecture and Products*, SG24-5593
- ▶ *The Three Steps to Super.Human. Software: Compare, Coexist, Migrate From Microsoft Exchange to Lotus Domino Part One: Comparison*, SG24-5614
- ▶ *The Three Steps to Super.Human. Software: Compare, Coexist, Migrate. From Microsoft Exchange to Lotus Domino. Part Two: Coexistence and Migration*, SG24-5615
- ▶ *A Roadmap for Deploying Domino in the Organization*, SG24-5617
- ▶ *Lotus Notes and Domino Take Center Stage: Upgrading from R4 to R5*, SG24-5630
- ▶ *Lotus Sametime Application Development Guide*, SG24-5651
- ▶ *IBM Enterprise Information Portal A Primer*, SG24-5749
- ▶ *Using Domino Workflow*, SG24-5963
- ▶ *Customizing QuickPlace*, SG24-6000
- ▶ *Using LDAP for Directory Integration: A Look at IBM SecureWay Directory, Active Directory, and Domino*, SG24-6163
- ▶ *Lotus Sametime 2.0 Deployment Guide*, SG24-6206
- ▶ *XML Powered by Domino How to use XML with Lotus Domino*, SG24-6207
- ▶ *Inside the Lotus Discovery Server*, SG24-6252
- ▶ *CommonStore for Domino, Exchange and SAP*, SG24-6405
- ▶ *Lotus Mobile and Wireless Solutions*, SG24-6525

Lotus Domino and IBM WebSphere Redbooks

- ▶ *Domino and WebSphere Together Second Edition*, SG24-5955
- ▶ *Applying the Patterns for e-business to Domino and WebSphere Scenarios*, SG24-6255

Other resources

These publications are also relevant as further information sources:

- ▶ *iSeries Performance Capabilities Reference Version 5 Release 1*, SC41-0607
- ▶ *OS/400 Work Management V4R5*, SC41-5306
- ▶ *Performance Tools for iSeries Version 5*, SC41-5340
- ▶ *BEST/1 Capacity Planning Tool for V4R1*, SC41-5341
- ▶ *CL Reference*, SC41-5722

Note, information related to OS/400 Version 5 Release 1 (V5R1) is now published in the iSeries Information Center, which is located on the Web at:

<http://publib.boulder.ibm.com/pubs/html/as400/infocenter.htm>

You can still obtain the V4R5 Work Management manual through the Information Center: Supplemental Manuals.

Referenced Web sites

These Web sites are also relevant as further information sources:

- ▶ Lotus Web site: <http://www.lotus.com>
- ▶ IBM home page: <http://www.ibm.com>
- ▶ Iris Cafe — Notes.Net: <http://www.notes.net>
- ▶ IBM Services: <http://www.ibm.com/services>
- ▶ IBM Software: <http://ibm.com/software>
- ▶ IBM @server iSeries home page: <http://ibm.com/servers/eserver/series>
- ▶ Lotus Domino on iSeries: <http://ibm.com/servers/eserver/series/domino>
- ▶ Domino for iSeries Performance:
<http://www.iseries.ibm.com/developer/domino/perform>
- ▶ Domino for iSeries Sizing:
<http://www-1.ibm.com/servers/eserver/series/domino/d4szintro.htm>
- ▶ IBM Workload Estimator for iSeries: <http://as400service.ibm.com/estimator>
- ▶ iNotes Web Access Performance Analysis white paper:
http://www.notes.net/today.nsf/lookup/inotes_performance
- ▶ iSeries Technical Studio: <http://www.iseries.ibm.com/tstudio/>
- ▶ iSeries Information Center:
<http://publib.boulder.ibm.com/pubs/html/as400/infocenter.htm>
- ▶ iSeries Online Library: <http://publib.boulder.ibm.com/pubs/html/as400/online1ib.htm>
- ▶ iSeries Performance Capabilities Reference:
<http://publib.boulder.ibm.com/html/as400/v5r1/ic2924/index.htm>
- ▶ PartnerWorld for Developers: <http://www.developer.ibm.com/>
- ▶ NotesBench Consortium: <http://www.notesbench.org>
- ▶ MidRange Performance Group: <http://www.mpginc.com/>
- ▶ Mercury Interactive: <http://www.merc-int.com/>
- ▶ Performance Management/400 (PM/400):
<http://www-1.ibm.com/services/its/us/mus14d1b.html>
- ▶ NotesBench Consortium: <http://www.notesbench.org>
- ▶ Technovations (GroupSizr WebSizr): <http://www.technovations.com>
- ▶ Computer Measurement Group (ARM): <http://www.cmg.org/regions/cmgarmlw>

How to get IBM Redbooks

You can order hardcopy Redbooks, as well as view, download, or search for Redbooks at the following Web site:

ibm.com/redbooks

You can also download additional materials (code samples or diskette/CD-ROM images) from that site.

IBM Redbooks collections

Redbooks are also available on CD-ROMs. Click the CD-ROMs button on the Redbooks Web site for information about all the CD-ROMs offered, as well as updates and formats.

Index

Symbols

#Waiters 137
\$UPDATEQUEUE 220
\$UpdateQueue 186
*ADD 230
*BASE 171
*BASE pool 168
*BASE storage pool 167
*BATCHJOB workload type 418
*CALC 169
*Calc 170
*CLS 275
*FIRST 230
*INTERACT 167, 169
*JOB 275
*JOBQ 275
*MACHINE pool 68, 165
*MGTCOL 83
*PROFILE 123
*SHRPOOL 168
*SPOOL 169
.DTF files 222
.sgf.notespartition 276
@DB commands 269
@DB functions
 on a Dedicated Server for Domino 354
@Now 190, 191
@Today 190, 191

Numerics

127.0.0.1 278
5722-PT1 12, 192
5769-PT1 192
5799-DSD 342

A

access control lists (ACL) 5
ACL 7
Actions page 94, 110
active users 24, 46
ActiveX 10
activity levels 170, 172
actuator 74
Add Environment Variable (ADDENVVAR) command 200
Add PEX Definition (ADDPEXDFN) command 116
ADDENVVAR (Add Environment Variable) command 200
ADDPEXDFN (Add PEX Definition) command 116
adjusting activity levels 172
ADLSVR 179
administration process (ADMINP) 232
ADMINP 233

ADMINP (administration process) 232
AdminPInterval 444
advanced Domino functions 57
After each use 190, 191
Agent Manager
 improving performance 237
 run multiple concurrently 237
Agent.Daily.AccessDenials 453
Agent.Daily.ScheduledRuns 453
Agent.Daily.TriggeredRuns 453
Agent.Daily.UnsuccessfulRuns 453
Agent.Daily.UsedRunTime 453
Agent.Hourly.AccessDenials 453
Agent.Hourly.ScheduledRuns 453
Agent.Hourly.TriggeredRuns 453
Agent.Hourly.UnsuccessfulRuns 453
Agent.Hourly.UsedRunTime 453
Algorithm for Recovery and Isolation Exploiting Semantics (ARIES) 386
AMGR 173, 225
 potential performance impacts 235
AMGR (Agent Manager) 234
AMgr_DocUpdate EventDelay 238
AMgr_DocUpdateAgentMinInterval 238, 239, 445
AMgr_DocUpdateEventDelay 239, 445
AMgr_NewMailAgentMinInterval 238, 239, 445
AMgr_NewMailEventDelay 238, 239, 445
AMgr_SchedulingInterval 238, 445
AMgr_UntriggeredMailInterval 238, 445
Analyze LOG.NSF 194
AnyMail/400 427
Apache 313
applet 10
application 67
application development performance 58
application hub server 292
Application Response Measurement (ARM) 16
application sizing 392
applied workload 392
archival logging style 384
archival policy legal advantages 209
archival TxL logs and 4 Gb (70 log) limit 387
ARIES (Algorithm for Recovery and Isolation Exploiting Semantics) 386
ARM (Application Response Measurement) 16
as400hlp.nsf 54
ASCII 202
ASCII tab delimited text 103
ASP 175, 373
 transaction logging files 175
ASP (auxiliary storage pool) 373
asynchronous I/O 382
Asynchronous I/O Completion Ports (IOCP) 63
Auto, after first use 190, 191
Auto, at most every.... 190, 191

- automatic failover 281
- Automatic option 191
- autostart
 - Domino with TCP/IP 239
 - TCP/IP during IPL 239
- Autostart entry, removing 225
- auxiliary storage 372
- auxiliary storage pool (ASP) 373
- availability 4
- AvgWait 137

B

- Backup Recovery and Media Services (BRMS) 178, 380, 384
- base Domino functions 56
- base pool 168
- batch job 189
- Batch Logical Database I/O 93
- batch pool 171
- BEST/1 13, 391
 - for Domino server capacity planning 399
 - for the AS/400 System 406
 - saving the model 423
 - workload components 414
- BILLING task 253
- BillingAddInRunTime 254
- BillingClass 253
- BillingWrite API 254
- Binary Large Objects (BLOB) 261
- BRMS 178, 275, 384
 - on a Dedicated Server for Domino 358
- BRMS (Backup Recovery and Media Services) 380
- BRMS/400 179
- BRMS/400 and how the QNNINBRM works 179
- BRMS/400 and performance 179
- buffer pool summary 215
- build a view 188
- business transaction 395
- BUSYTIME.NSF 252

C

- CALCONN 255
- CALCONN (Calendar connector) 254
- Calendar & Scheduling 397
- Calendar.Total.All.Appointments.Reservations 453
- Calendar.Total.All.Users.Resources 453
- Calendar.Total.Appts 453
- Calendar.Total.Reservations 453
- Calendar.Total.Resources 453
- Calendar.Total.Users 453
- calibrating response times 421
- calibration expertise 418
- Call Level Interface (CLI) 269, 270, 350
- capacity planning 392
 - basic process 392
 - CPW values 393
 - prerequisites 392
 - unit of measure 395
- capacity planning unit of measure (CPUM) 395, 401

- capacity sizing 392
- capturing NRPCs 157
- CCCVS (Conversion Services) 428
- CCSDI set to wrong character set causing performance problems 204
- CFGDOMSVR (Configure Domino Server) command 276
- Change Environment Variable (CHGENVVAR) command 200
- Change Job Type (CHGJOBTYP) command 115
- Change Shared Pool (CHGSHRPOOL) command 168
- Change Subsystem Description (CHGSBSD) command 168, 169
- checkpoint 388
 - for transaction logging 378
- checkpoint frequencies 378
- CHGDOMSVR 179
- CHGENVVAR (Change Environment Variable) command 200
- CHGIPLA 239
- CHGJOBTYP (Change Job Type) command 115
- CHGSBSD (Change Subsystem Description) command 168, 169
- CHGSHRPOOL (Change Shared Pool) command 168
- CHGSYSLIBL 131, 132
- Chronos 219
- circular transaction logging 384
- CIW (Compute Intensive Workload) 45, 394
- CLADMIN (Cluster Administration Process) 245
- class 174, 275
- class description 64
- classifying jobs into workloads 409
- CLDBDIR (Cluster Database Directory Manager) 246
- CLEANUP 79
- CleanupScriptPath=call qnotes/nsd 17
- CleanupTimeout 436
- CLI (Call Level Interface) 269, 350
- Client_Clock 157
- CLREPL (Cluster Replicator) 246
- CLREPL_Poll_Interval 451
- Cluster Administration Process (CLADMIN) 245, 285
- cluster analysis tool 296
- Cluster Database Directory (CLBDIR.NSF) 286
- Cluster Database Directory Manager 246
- cluster management software 279
- Cluster Manager 285
- cluster node, switching and restarting Domino 281
- cluster replication 27
- Cluster Replicator (CLREPL) 285
- Cluster Replicator tuning 296
- cluster statistics cache 286
- Cluster unaware browser client 285
- Cluster_Replicators 451
- Cluster-aware Notes client 285
- clustering 27, 57, 278, 398
 - Domino servers 283
 - mail.box not recommended 291
 - on the same iSeries server 27
 - tips 297
- ClusterProven Domino 282

- ClusterProven Domino for iSeries 280
- COLLECT task 246
- COLLECT_DB_LOCK_WAITS=1 137
- collecting performance data 83
- Collection Interval 109
- Collection retention period 102
- Collection Services 12, 53, 83, 85, 88, 105
- Collector server task 151
- Collector.Time.Collecte d 453
- Collector.Time.Elapsed 453
- combining metrics 94, 102
- Comm.NumOldSessionsClosed 454
- comma separated variable 103
- commands 127
- Commercial Processing Workload (CPW) 348, 393
- common interface architecture 280
- common name (CN) 5
- Common Object Request Broker Architecture (CORBA) 270
- Communication IOP Utilization (Average) 93, 104
- Communication IOP Utilization (Maximum) 93
- Communication Line Utilization (Average) 93
- Communication Line Utilization (Maximum) 94
- communications 67
 - I/O rate 108
- Communications I/O rate 108
- COMPACT 385
- compact 205
- COMPACT -b 205
- COMPACT -b nightly 206
- COMPACT task 256
- complementary processing 346
- complementary workload 342, 344, 369
- Compute Intensive Workload (CIW) 45, 349, 394
- concurrency rate 24
- concurrent users 24, 26
- concurrently active users 46
- configuration document 185, 436
- Configure Domino Server (CFGDOMSVR) command 276
- Confirm Creation of BEST/1 Model window 413
- connection documents referencing the Loopback port 199
- console logging 200
- Console_Loglevel 444
- continuation reference 331
- continuous availability market 279
- Convert Performance Thread Data 116
- CORBA 227
- CORBA (Common Object Request Broker Architecture) 270
- CPF0908 171
- CPF0909 171
- CPU breakdown 393
- CPU utilization 107, 108
- CPU Utilization (Average) 91, 104
- CPU Utilization (Database Capability) 92, 104
- CPU Utilization (Interactive Feature) 92, 104
- CPU Utilization (Interactive Jobs) 91, 104
- CPU Utilization (Secondary Workloads) 92, 105

- CPU Utilization Basic (Average) 92
- CPU Utilization metrics 104
- CPUM (capacity planning unit of measure) 395, 401
- CPW 44, 47
- CPW values 393
- Create BEST/1 Model from Performance Data window 408
- Create BEST/1 Model window 407
- Create Class (CRTCLS) command 174
- Create User-Defined FS (CRTUDFS) command 376
- creating a model using measured data 404
- CRM (Customer Relationship Management) 263
- CRTCLS (Create Class) command 174
- CRTPFRDTA 83
- CRTUDFS (Create User-Defined FS) command 376
- cursor 261
- Customer Relationship Management (CRM) 263
- CVTPFRTHD (Convert Performance Thread Date) command 116

D

- DASD 21
 - system ASP, user ASP, independent ASP 175
- DASD (direct access storage device) 372
- data notes 187
- data space 261
- Database Capability 92, 363
- Database Capacity 40
- database hardening 388
- Database Instance ID (DBIID) 205, 377, 385
- database replication 398
- Database.Buffer 216
- Database.BufferControlPool.Peak 454
- Database.BufferControlPool.Size 454
- Database.BufferControlPool.Used 454
- Database.BufferPool 135
 - MM.Reads 135
 - Peak.megabytes 135
 - PerCentReadsInBuffer 134, 135
- Database.BufferPool.Allocated 454
- Database.BufferPool.Maximum 212, 454
- Database.BufferPool.Maximum.Megabytes 135
- Database.BufferPool.MM.Writes 135
- Database.BufferPool.Peak 212, 454
- Database.BufferPool.PerCentReadsInBuffer 454
- Database.BufferPool.Reads 454
- Database.BufferPool.Used 212, 454
- Database.BufferPool.Writes 454
- DataBase.DbCache 216
- DataBase.DbCache.CurrentEntries 454
- DataBase.DbCache.HighWaterMark 454
- DataBase.DbCache.Hits 454
- DataBase.DbCache.InitialDbOpens 454
- DataBase.DbCache.Lookups 454
- DataBase.DbCache.MaxEntries 454
- DataBase.DbCache.OvercrowdingRejections 454
- Database.ExtMgrPool 216
- Database.ExtMgrPool.Peak 454
- Database.ExtMgrPool.Used 454
- Database.NIFPool.Size 454

- Database.NIFPool.Used 454
- Database.NSF.FreeHandleStack.FreeHandleStackHits 454
- Database.NSF.FreeHandleStack.HandleAllocations 454
- Database.NSF.FreeHandleStack.MissRate 454
- Database.NSFPool.Peak 454
- Database.NSFPool.Size 454
- Database.NSFPool.Used 454
- Database.RM.Current.K.Restart.Redo.Hold 454
- Database.RM.Current.K.UnCommitted.Undo.Hold 454
- Database.RM.Current.MaxChkptScan.msec 454
- Database.RM.LastChkpt.Avg.K.Logged.PerTran 454
- Database.RM.LastChkpt.Interval.Sec 455
- Database.RM.LastChkpt.K.Logged 455
- Database.RM.LastChkpt.Time 455
- Database.RM.LastChkpt.Total.DBs 455
- Database.RM.LastChkpt.Trans 455
- Database.RM.Peak.Avg.K.Logged.PerTran 455
- Database.RM.Peak.Interval.Sec 455
- Database.RM.Peak.K.Logged 455
- Database.RM.Peak.Time 455
- Database.RM.Peak.Total.DBs 455
- Database.RM.Peak.Trans 455
- Database.RM.Peak.Trans.PerMin 455
- Database.RM.Restart.Duration.Sec 455
- Database.RM.Restart.K.Applied 455
- Database.RM.Restart.Time 455
- Database.RM.SinceChkpt.K.Logged 455
- Database.RM.SinceChkpt.Trans 455
- Database.RM.SinceStartup.Aborts 455
- Database.RM.SinceStartup.Critical.Log.Times 219, 455
- Database.RM.SinceStartup.Log.Recs.Analyzed 455
- Database.RM.SinceStartup.Log.Recs.Applied 455
- Database.RM.SinceStartup.Log.Recs.Read 455
- Database.RM.SinceStartup.Log.Recs.Written 455
- Database.RM.SinceStartup.M.Logged 455
- Database.RM.SinceStartup.Trans 455
- Database.RM.Sys.Log.Type 455
- Database.RM.Sys.Logged 455
- Database.RM.Sys.M.Chkpt.Interval 456
- Database.RM.Sys.M.Log.Size 456
- Database.RM.Sys.M.Redo.Limit 456
- Database.RM.Sys.Phase 456
- DB CPU Util column 72
- DB2 processing
 - on a Dedicated Server for Domino 344
 - shown in WRKSYSACT 72
- DB2 UDB 261
 - integrating with Domino 263
- DB2Connect 269
- DbDir_Refresh_Interval 436
- DBID 385
- DBIID
 - changed by compact 385
 - changes 385
- DBIID (Database Instance ID) 205, 377, 385
- DBMS 261
- debug 129
- Debug_AMgr 235
- DEBUG_AMGR=n 235
- Debug_Console 444
- DEBUG_DISABLE_FAIRSEM 223
- DEBUG_OS400_PEX 117
- DECS 264
 - as component of LEI 354
 - workload scenarios for DSD 352
- DECS (Domino Enterprise Connection Services) 351
- Dedicated Server for Domino (DSD) 44, 341, 342
- Dedicated System Tools (DST) 374
- Default_Index_Lifetime_Days 244, 446
- Define Non-Interactive Transactions window 412
- developing a naming structure 5
- different workloads for Domino 396
- direct access storage device (DASD) 372
- directory synchronization 2, 255
 - removing 225
- Disable_Cluster_Replicator 451
- disaster preparedness 284
- Discard Index 187, 190, 191
- disconnected users 26
- disk 67
- Disk Arm Utilization (Average) 93, 104
- Disk Arm Utilization (Maximum) 93
- Disk Drives 41
- Disk I/O rate 108
- Disk IOP 380
- Disk IOP Utilization (Average) 93
- Disk IOP Utilization (Maximum) 93
- disk performance 393
- disk pool 373
- disk statistics 141
- Disk Storage (Average) 93
- Disk Storage (Maximum) 93
- Display Call Stack 131
- Display System Log (DSPLOG) command 81
- Distributed Relational Database Architecture (DRDA) 345
 - on Dedicated Server for Domino 355
- dividing CPU utilization 61
- doclinks 10
- document unique ID 263
- document-centric object store 10
- DOLS 314
- Domino
 - advanced functions 57
 - base functions 56
 - Calendar & Scheduling 397
 - commands 127
 - jobs 231
 - logging 223
 - mail concepts 28
 - mail user 397
 - statistical reports 127
 - switching and restarting on different cluster nodes 281
 - tasks on iSeries 57
 - trace points 117
 - transactions 56
 - tuning iSeries performance 161, 183
 - user applications 397

- Web serving 397
- Domino 5 Administration Help 54
- Domino application design tips and techniques 259
- Domino architecture 9
- Domino cluster 284
 - benefits summary 283
 - defining, starting Domino on multiple nodes 281
- Domino clustering 27, 278, 283
 - limitations 290
 - on iSeries and across all platforms 227
 - planning 289
 - versus ClusterProven Domino for iSeries 282
- Domino console logging 200
- Domino console platform 142
- Domino Directory 11, 436
- Domino Enterprise Connection Services (DECS) 264, 351
- Domino for iSeries
 - available tools 11
 - Help 54
 - performance tests with TxL 381
 - planning 5
 - sizing using the Workload Estimator 19
 - understanding 56
- Domino HTTP Clustering
 - components and technology with ICM 287
 - ICM and bookmarks 288
 - ICM characteristics of 287
 - ICM corporate and enterprise use of 287
 - ICM platform support 287
 - supported ICM network protocols 288
- Domino HTTP server best practices 310
- Domino HTTP server partitioning 311
- Domino Indexer 185
- Domino Internet Cluster Manger (ICM) 314
- Domino Off-Line Services (DOLS) 314
- Domino panel group 203
- Domino parameters that affect performance 185
- Domino partitioning 274, 275
 - planning 276
 - tips 277
- Domino performance methodology for iSeries 54
- Domino server
 - clustering 283
 - communication between servers on the same iSeries server 278
 - partitioning 274
- Domino server database housekeeping jobs 256
- Domino server environment 401
- Domino server jobs 230
- Domino server jobs, always necessary 232
- Domino server jobs, very likely not used 253
- Domino server managed like any other cluster resource. 282
- Domino server modeling 400
- Domino server workload 396
- Domino task threads 64
- Domino tasks as they relate to AS/400 jobs 184
- Domino Web Server API (DSAPI) 314
- Domino workload 24, 166

- Domino.BuildName 456
- Domino.BuildNumber 456
- Domino.BuildPlatform 456
- Domino.BuildVersion 456
- Domino.Command.CreateDocument 456
- Domino.Command.DeleteDocument 456
- Domino.Command.EditDocument 456
- Domino.Command.Login 456
- Domino.Command.Navigate 456
- Domino.Command.OpenAbout 456
- Domino.Command.OpenAgent 456
- Domino.Command.OpenDatabase 456
- Domino.Command.OpenDocument 456
- Domino.Command.OpenElement 456
- Domino.Command.OpenForm 456
- Domino.Command.OpenHelp 456
- Domino.Command.OpenIcon 456
- Domino.Command.OpenNavigator 457
- Domino.Command.OpenServer 457
- Domino.Command.OpenView 457
- Domino.Command.ReadForm 457
- Domino.Command.SearchSite 457
- Domino.Command.Total 457
- Domino.Command.Unknown 457
- Domino.Config 135, 136
 - ActiveThreads.Max 135
 - ActiveThreads.Min 135
- Domino.Requests.Per1Day.Peak 457
- Domino.Requests.Per1Day.PeakTime 457
- Domino.Requests.Per1Day.Total 457
- Domino.Requests.Per1Hour 457
- Domino.Requests.Per1Hour.Peak 457
- Domino.Requests.Per1Hour.PeakTime 457
- Domino.Requests.Per1Hour.Total 457
- Domino.Requests.Per1Minute 457
- Domino.Requests.Per1Minute.Peak 457
- Domino.Requests.Per1Minute.PeakTime 457
- Domino.Requests.Per1Minute.Total 457
- Domino.Requests.Per5Minute 457
- Domino.Requests.Per5Minute.Peak 457
- Domino.Requests.Per5Minute.PeakTime 457
- Domino.Requests.Per5Minute.Total 457
- Domino.Requests.Total 457
- Domino.Threads
 - Active.Peak 136
 - Peak.Time 136
- Domino.Threads.Active.Peak 26
- Domino.Threads.Peak.Total 136
- domino_classes 174
- DominoAnalyzeFormulas 446
- DominoAsynchronizeAgents 238, 445
- Domino-based applications 343, 345
- DRDA 345, 350
 - on a Dedicated Server for Domino 355
- DRT (Delivery Report Task) 428
- DSAPI 314
- DSD (Dedicated Server for Domino) 341
- DSPLOG (Display System Log) command 81
- DST 374
- DTF file extension 222

Dump utility 17
Dynamic @Function agents 311

E

EBCDIC 202
EDC 199, 333
EDC compared to standard DirCat 199
Edit File Utility 174
Edit Job Classifications window 410
EDTF 174
EnableJavaAgents 225, 437
End Performance Explorer (ENDPEX) 116
Ending Collection Services 89
ENDJOB 239
ENDPEX (End Performance Explorer) 116
ENDPEX (End PEX) command 117
Enterprise JavaBeans (EJBs) 357
Enterprise Resource Planning (ERP) 263
environment variable settings (iSeries) 224
ERP (Enterprise Resource Planning) 263
Error 500 317
error recovery procedure 82
ESMTP 253
Ethereal 156
Ethernet port filtering 180
Event server task 152
EVENT task 247
Event task 150
event-driven cluster replication 284
events 145
EVENTS4.NSF 247
events4.nsf 132, 137
excessive translations on iSeries and zSeries 202
execution control lists (ECL) 5
expert cache 168, 169
Export Graph History to PC 103
Extended DirCat 199
Extended Directory Catalog 199
Extended Directory Catalog to improve performance 200

F

failover 57
FairReadWriteSemaphores 223
Faster backups, reduce volumes, increase recoverability 179
faulting rates 171
Favor Restart 378
Favor Restart Recovery Time 378
Favor Runtime 378
FD (file descriptor) 137
file descriptor (FD) 137
firewall 5
first-in, first-out priority queue 70
FIXUP 379
Fixup 437
Fixup task 257
Fixup_Tasks 437
flexibility 284
flushing 377

FRCRATIO 373
FRWSem 223
FT_Alternate_Filter 446
FT_Domain_Directory_Name 446
FT_DOMAIN_IDXTHDS 446
FT_HTML_Title 446
FT_Index_Accent_Sens 446
FT_Index_Attachments 446
FT_Index_Ignore_Attachment_Types 446
FT_Intl_Setting 446
FT_KV_LOG 446
FT_LIBName 447
FT_Max_Instances 447
FT_Max_Instances_Large 447
FT_Max_Search_Results 447
FT_No_CompWinTitle 447
FT_Summ_Default_Language 447
FT_Use_AltFtr 448
FTG_Index_Limit 448
FTG_No_Summary 448
FTV_Fields_ 448
FTV_Max_Fields 448
full backup 178, 384
full text indexes 189
FullTextMultiProcess 220, 448

G

general notes on sizing pools 172
Get_Partition_Number API 276
Global Text Retrieval (GTR) 199
GO PERFORM 86
Graph History performance data 102
GroupSizr 15
GTR 199
GTR search engine 199
GWVJOB 131

H

hardening 377
hardening databases and logfiles 388
help5_admin.nsf 54
hierarchical naming 5
high availability of critical applications 283
highly available applications using a common interface architecture 280
highly available SMTP gateways 291
hit rates 33
how databases are removed from the cache 222
HSL OptiConnect 280
HTML file request 304
HTML translation 304
HTTP 57
HTTP 1.0 315
HTTP 1.1 315
HTTP hit rate 33
 calculate 33
HTTP performance tuning 302
HTTP server 226, 304
HTTP server statistics 135

HTTP task 305
HTTPSETUP 211

I

I/O completion ports (IOCP) 243, 305
IASP (independent ASP) 175
IBM Business Partner cluster management software 279
IBM Toolbox for Java 228
IBM Workload Estimator for iSeries (WLE) 20
ICM 314
ICM and bookmarks with Domino HTTP Clustering 288
ICM and Domino 289
ICM and Domino HTTP Clustering
 components and technology 287
 functional summary 287
ICM and SSL 288
ICM and user identification plus authorization 288
ICM characteristics of Domino HTTP Clustering 287
ICM conditions triggering failover and load balancing to occur 288
ICM corporate and enterprise use of Domino HTTP Clustering 287
ICM Domino HTTP Clustering performance 289
ICM network protocols supported by Domino HTTP Clustering 288
ICM non-targeted requests 289
ICM platform support for Domino HTTP Clustering 287
ICM server availability index and maximum scalability 288
ICM unauthorized access 288
ICM URL processing 289
ICM workload balancing 288
ICM.Command.Redirects.ClusterBusy 457
ICM.Command.Redirects.Successful 457
ICM.Command.Redirects.Unsuccessful 457
ICM.Command.Total 457
ICM.Command.Unknown 457
ICM.Receive.Error 457
ICM.Requests.Per1Hour.Total 457
ICM.Requests.Per1Minute.Total 458
ICM.Requests.Per5Minutes.Total 458
ICM.Sessions.Threads.Busy 458
ICM.Sessions.Threads.Idle 458
ICM.Sessions.Total 458
iDoctor 121
iDoctor for iSeries 122
IFS 373
IMAP 26, 327
IMAP server 226, 327
IMAP.Sessions.Accept.Queue 458
IMAP.Sessions.Active 458
IMAP.Sessions.Inbound.BytesReceived 458
IMAP.Sessions.Inbound.BytesSent 458
IMAP.Sessions.Inbound.Non-SSL 458
IMAP.Sessions.Inbound.SSL 458
IMAP.Sessions.Inbound.SSL.Bad_Handshake 458
IMAP.Sessions.Inbound.Total 458
IMAP.Sessions.Peak 458
IMAP.Sessions.Threads.Busy 458
IMAP.Sessions.Threads.Idle 458
IMAP.Sessions.Total 458
IMAP4 319
IMAPMaxSession 226
IMSGCNV (Inbound Message Conversion) task 428
inappropriate processing 346
incremental backup 384
independent ASP (IASP) 175
index refresh option 191
Indexer 185, 186, 244
indexing and associated costs 214
indices and views not TxL logged 389
ineligible queue 70
initial thread 63
iNotes 337
 remote access 292
iNotes Access for MS Outlook 337
iNotes Access for Outlook 26
iNotes Web Access
 client requirements 339
 performance analysis 480
instance 24
Integrated File System (IFS) 260, 373
Integrated xSeries Server on Dedicated Server for Domino 357
integration 4
Interactive 171
interactive 173
interactive applications 167
interactive AS/400 tuning chart 164
Interactive CPW 173, 349
Interactive Response Time (Average) 92
Interactive Response Time (Maximum) 92
Internet address
 local 78
 remote 79
IOP 13, 67
IP address takeover to a new cluster 281
IPL 239
iSeries clustering 279
iSeries Continuous availability 279
iSeries Online Library (URL) 480
iSeries parameter TCPKEEPALV 218
iSeries server
 environment variable settings 224
 excessive translation 202
 synchronizing and managing multiple 279
 threads 63
iSeries solutions for the continuous availability market 279
iSeries Technical Studio (URL) 480
iSeries Toolkit for Java 355
iServer backups using BRMS/400 179
ISESCTL task 429
ISESHL 430
ISpy RunJava 226

J

Java 261
 on a Dedicated Server for Domino 356
Java Database Connectivity (JDBC) 270

- on a Dedicated Server for Domino 355
- Java servlet 227, 343
- Java Toolbox for AS/400 (JT400.JAR) 228
- JavaServer Pages (JSPs) 357
- JavaUserClasses 228
- JDBC (Java Database Connectivity) 270, 355
- job 63
 - run attributes of 64
- job class 176
- Job Count 107
- job description 275
- Job Log Message 107
- job monitor 13, 90, 105
 - log event 110
 - open event log 110
 - open monitor 110
 - run OS/400 command 110
 - starting 112
- job monitoring 105
- job numeric values 107
- job priority 70
 - considerations 66
- job queue 275
- Job Status 107
- Jobs to monitor 106
- journal receivers 374
- JPING 75
- JT400.JAR 228

K

- kernel 10
 - extension 10
 - thread 63
- key/think wait 70

L

- LAN Utilization (Average) 94
- LAN Utilization (Maximum) 94
- large file attachments 311
- layered end-user applications 11
- LC LSX (Lotus Connector LotusScript Extension) 270
- LC LSX (Lotus Connector LotusScript Software Extension) 354
- LDAP (Lightweight Directory Access Protocol) server 330
- LDAP server 226
- LDAP.Anonymous LDAP Connections 458
- LDAP.Average LDAP Search time 458
- LDAP.Failed LDAP Connections 458
- LDAP.Longest LDAP Search request 458
- LDAP.Longest LDAP Search time 458
- LDAP.Sessions.Accept.Queue 458
- LDAP.Sessions.Active 458
- LDAP.Sessions.Inbound.BytesReceived 458
- LDAP.Sessions.Inbound.BytesSent 458
- LDAP.Sessions.Inbound.Non-SSL 458
- LDAP.Sessions.Inbound.SSL 458
- LDAP.Sessions.Inbound.SSL.Bad_Handshake 458
- LDAP.Sessions.Inbound.Total 458

- LDAP.Sessions.Peak 459
- LDAP.Sessions.Threads.Busy 459
- LDAP.Sessions.Threads.Idle 459
- LDAP.Sessions.Total 459
- LDAP.Simple LDAP Connections 459
- LDAP.Strong Authentication Connections 459
- LDAP.Total LDAP Abandons 459
- LDAP.Total LDAP Adds 459
- LDAP.Total LDAP Compares 459
- LDAP.Total LDAP Connections 459
- LDAP.Total LDAP Deletes 459
- LDAP.Total LDAP Extended Operations 459
- LDAP.Total LDAP Modifies 459
- LDAP.Total LDAP ModifyDNs 459
- LDAP.Total LDAP Searches 459
- LDAP.Total LDAP Searches for Root DSE 459
- LDAP.Total LDAP Searches for Subschema 459
- LEI 263, 264
 - improve performance 269
- LEI (Lotus Enterprise Integrator) 353
- Level RSC 344
- Licensed Internal Code (Level RSC) 344
- Licensed Internal Program Interface (LIPI) 350
- Linux on Dedicated Server for Domino 361
- LIPI (Licensed Internal Program Interface) 350
- LMBCS (Lotus Multi-Byte Character Set) 202
- LoadRunner 11, 15
- local area network (LAN) 179
- locked views 188
- lockSlowWithInitAllowSig__9Qp0wMutexFv 203
- LockWaits 137
- Log <parameter> 442
- log analysis tool 295
- Log event 94
- LOG.NSF 128
- log.nsf 145, 146, 192, 223
- LOG_AGENTMANAGER 223
- Log_AgentManager 444
- Log_Disable_Session_Info 444
- LOG_MAILROUTING 224
- Log_MailRouting 444
- Log_Replication 442
- Log_Tasks 443
- LOG_UPDATE 192, 193
- Log_Update 443
- Log_View_Events 443
- LOGASIO job 249
- logfile hardening 388
- logging replication 295
- logging style 384
- logging with Domino 223
- Logical I/O rate 107, 108
- logical partitioning (LPAR) 274, 360
- long wait 69, 70
- loopback interface 199, 278
- LOOPBACK_TcpIPAddress 199
- Lotus 1-2-3 103
- Lotus Client NRPC Monitoring Tool 157
- Lotus Connector LotusScript Extension (LC LSX) 270
- Lotus Connector LotusScript Software Extension (LC

- LSX) 354
- Lotus Domino for AS/400 1, 3
 - overall performance 59
- Lotus Enterprise Integrator 263, 264
- Lotus Enterprise Integrator (LEI)
 - on a DSD 353
- Lotus Multi-Byte Character Set (LMBCS) 202
- Lotus Quality Engineer 17
- LotusScript 10, 261
- LotusScript agent 311
- LotusScript:Data Object (LS:DO) on a DSD 354
- LPAR 26, 81
 - on a DSD 360
- LPAR (logical partitioning) 274
- LS:DO
 - on a DSD 354
- LS:DO on a DSD 354

M

- machine pool 168
- Machine Pool Faults (Average) 93
- Mail and Calendaring Users (MCU) 46, 348, 394
- mail archival policy 209
- mail archiving 208
- mail backlogs and performance issues 200
- mail concepts 28
- mail database integration 291
- mail routing 240
- mail server failover 292
- mail trapped in mail.box 291
- mail user 397
- Mail.AverageDeliverTime 459
- Mail.AverageServerHops 459
- Mail.AverageSizeDelivered 459
- mail.box 240
- Mail.DBCacheEntries 212
- Mail.DBCacheHits 213
- Mail.DBCacheReads 213
- Mail.Dead 459
- Mail.Delivered 459
- Mail.Deliveries 459
- Mail.Domain 459
- Mail.Hold 459
- Mail.IMAP.Cache.MsgInsertAttempts 459
- Mail.IMAP.FindCacheAttempted 459
- Mail.IMAP.NumCacheHits 459
- Mail.IMAP.NumMsgInsertedInCache 459
- Mail.MaximumDeliverTime 459
- Mail.MaximumServerHops 459
- Mail.MaximumSizeDelivered 459
- Mail.MinimumDeliverTime 460
- Mail.MinimumServerHops 460
- Mail.MinimumSizeDelivered 460
- Mail.PeakByteTransferRate 460
- Mail.PeakMessagesTransferred 460
- Mail.PeakMessageTransferRate 460
- Mail.PeakMessageTransferRate.Time 460
- Mail.PeakTotalBytesTransferred 460
- MAIL.TotalFailures 460
- Mail.TotalKBTransferred 460

- Mail.TotalPending 460
- Mail.TotalRouted 460
- Mail.TotalRouted.SMTP 460
- Mail.TotalRouter.NRPC 460
- Mail.TransferFailures 460
- Mail.Transferred 460
- Mail.Waiting 460
- Mail.WaitingforDNS 460
- Mail.WaitingRecipients 460
- Mail_Log_To_MiscEvents 443
- Mail_Number_of_Mail_Boxes 450
- Mail_Transfer_Retry_Minutes 450
- Mailbox quotas 208
- MailByDest.<destServerName>.Delivered 460
- MailByDest.<destServerName>.TotalFailures 460
- MailByDest.<destServerName>.TotalRouted 460
- MailClusterFailover 449
- MailCompactDisabled 449
- MailCompactDisabled= 257
- MailDisablePriority 449
- Mailfile size 209
- MailLowPriorityTime 449
- MailMaxConcurrentXferThreads 449
- MailMaxDeliveryThreads 449
- MAILMAXTHREADS 225
- MailMaxThreads 450
- main storage 67, 372, 393
- Make Directory (MD) command 376
- Management Central 12, 13, 89, 192
 - APIs 88
 - CPU Utilization metrics 104
 - performance monitor 89
- manual calibration of resource usage 420
- maximum cached commands 308
- maximum cached designs 309
- maximum cached users 309
- maximum log file sizes 387
- maximum number of users 201, 275
- Maximum requests over a single connection 306
- maximum transmission unit (MTU) 179
- Maximum Transmission Unit Size (MTU)
 - list of recommended values 180
- MaxRetryDelay 450
- MaxWaiters 137
- MCU 28, 45, 46, 48, 394
- MCU (Mail and Calendaring Users) 394
- MD (Make Directory) command 376
- measuring inbound SMTP performance 323
- measuring SMTP performance 320
- measuring tuning thread activity 307
- Mem.Allocated 460
- Mem.Allocated.Process 460
- Mem.Allocated.Shared 460
- Mem.Availability 460
- Mem.Free 460
- Mem.FTVDKMaxBufferSizeUsed 460
- Mem.PhysicalRam 460
- Mem.Quota 460
- Mem.SwapFile.Size 461
- Mem.Timeouts 461

- memory pool 166, 167
- Memory statistics 140
- memory statistics 140
- Memory_Quota 437
- message queue for QSYSOPR 171
- Message Transfer Agent 11
- Metrics page 91, 107
- MF27217 345
- MHz 46
- Microsoft Excel 97 103
- MinNewMailPoll 450
- mirrored disk 175
- mirrored storage protection (RAID0) 374
- Mod 137
- moderate mail user 46
- modifying an existing system monitor 96
- Monitor.<EventName>.Failure 461
- Monitor.<EventName>.Fatal 461
- Monitor.<EventName>.Warning(High) 461
- Monitor.<Taskname>.Heartbeat 461
- monitoring performance with iSeries tools 53
- MTA 11, 430
- MTA.ccMail.Dead 461
- MTA.ccMail.TotalKBTransferred 461
- MTA.ccMail.TotalRouted 461
- MTA.ccMail.TransferFailures 461
- MTA.ccMail.Transferred 461
- MTA.ccMail.Waiting 461
- MTA.ccMail.WaitingRecipients 461
- MTA.Smtp.Dead 461
- MTA.Smtp.TotalKBTransferred 461
- MTA.Smtp.TotalRouted 461
- MTA.Smtp.TransferFailures 461
- MTA.Smtp.Transferred 461
- MTA.Smtp.Waiting 461
- MTA.Smtp.WaitingRecipients 461
- MTAWQ 430
- MTC 227
- MTS 430
- MTU 179, 180
 - list of recommended values 180
- MTU (maximum transmission unit) 179
- multiple databases 398
- multiple Indexer tasks on the server 220
- multiple IP addresses 276
- multiple iSeries servers 279

N

- Names and Address Book 11
- NAMES.NSF 240
- native HTMP pages 311
- NET 461
 - NET.<PortName>.BytesReceived 461
 - NET.<PortName>.BytesSent 461
 - NET.<PortName>.Sessions.Established.Incoming 461
 - NET.<PortName>.Sessions.Established.Outgoing 461
 - NET.<PortName>.Sessions.Limit 461
 - NET.<PortName>.Sessions.LimitMax 461
 - NET.<PortName>.Sessions.LimitMin 461
 - NET.<PortName>.Sessions.Peak 462
- NET.<PortName>.Sessions.Recycled 462
- NET.<PortName>.Sessions.Recycling 462
- NET.Log.<username>.UnwrittenEntries 462
- NET.TCP.BytesReceived 462
- NET.TCP.BytesSent 462
- NET.TCP.Sessions.Established.Incoming 462
- NET.TCP.Sessions.Established.Outgoing 462
- NET.TCP.Sessions.Limit 462
- NET.TCP.Sessions.LimitMax 462
- NET.TCP.Sessions.LimitMin 462
- NET.TCP.Sessions.Peak 462
- NET.TCP.Sessions.Recycled 462
- NET.TCP.Sessions.Recycling 462
- NET.TCPIP.BytesReceived 462
- NET.TCPIP.BytesSent 462
- NET.TCPIP.Sessions.Established.Incoming 462
- NET.TCPIP.Sessions.Established.Outgoing 462
- NET.TCPIP.Sessions.Limit 462
- NET.TCPIP.Sessions.LimitMax 462
- NET.TCPIP.Sessions.LimitMin 462
- NET.TCPIP.Sessions.Peak 462
- NET.TCPIP.Sessions.Recycled 462
- NET.TCPIP.Sessions.Recycling 462
- NET.VINES.BytesReceived 462
- NET.VINES.BytesSent 462
- NetServer on a DSD 358
- NETSTAT (Work with TCP/IP Network Status) command 75
- network performance 74, 292
- network sniffer 156
- next hop 77
- NFS_Buffer_Pool_Size 437
- NFS_Buffer_Pool_Size_MB 437
- NIF 187
- NIF (Notes Indexing Facility) 185
- NIF-semaphores 223
- NNTP (Network News Transport Protocol) server 334
- NNTP.<RemoteServerName>.Articles.Posted 335, 462
- NNTP.<RemoteServerName>.Articles.Sent 335, 462
- NNTP.<RemoteServerName>.Bytes.Received 335, 462
- NNTP.<RemoteServerName>.Bytes.Sent 335, 462
- NNTP.<RemoteServerName>.Pull.Articles.Failed 335, 462
- NNTP.<RemoteServerName>.Pull.Articles.Offered 335, 463
- NNTP.<RemoteServerName>.Pull.Articles.Requested 335, 463
- NNTP.<RemoteServerName>.Pull.Articles.Transferred 335, 463
- NNTP.<RemoteServerName>.Push.Articles.Failed 335, 463
- NNTP.<RemoteServerName>.Push.Articles.Offered 335, 463
- NNTP.<RemoteServerName>.Push.Articles.Requested 335, 463
- NNTP.<RemoteServerName>.Push.Articles.Transferred 335, 463
- NNTP.Articles.Posted 335, 463
- NNTP.Articles.Sent 335, 463
- NNTP.Bytes.Received 335, 463

- NNTP.Bytes.Sent 335, 463
- NNTP.Pull.Articles.Failed 335, 463
- NNTP.Pull.Articles.Offered 335, 463
- NNTP.Pull.Articles.Requested 335, 463
- NNTP.Pull.Articles.Transferred 335, 463
- NNTP.Push.Articles.Failed 335, 463
- NNTP.Push.Articles.Offered 336, 463
- NNTP.Push.Articles.Transferred 336, 463
- NNTP.Sessions.Accept.Queue 336, 463
- NNTP.Sessions.Active 336, 463
- NNTP.Sessions.Inbound.BytesReceived 336, 463
- NNTP.Sessions.Inbound.BytesSent 336, 463
- NNTP.Sessions.Inbound.Non-SSL 336, 463
- NNTP.Sessions.Inbound.SSL 336, 464
- NNTP.Sessions.Inbound.SSL.Bad_Handshake 336, 464
- NNTP.Sessions.Inbound.Total 336, 464
- NNTP.Sessions.Outgoing.non-SSL 336, 464
- NNTP.Sessions.Outgoing.SSL 336, 464
- NNTP.Sessions.Peak 336, 464
- NNTP.Sessions.Threads.Busy 336, 464
- NNTP.Sessions.Threads.Idle 336, 464
- NNTP.Sessions.Total 336, 464
- No_Force_Activity_Logging 443
- No_Force_Activity_Logging=1 244
- NoMsgCache 450
- non-Domino applications 399
- note 10
- note.ini 237
- Notes Agent Builder 234
- Notes API 285
- Notes client cluster components 285
- Notes Client Mail access 26
- Notes DB Request 304
- Notes Indexing Facility (NIF) 185, 187
- Notes log database 145
- Notes naming structure 5
- Notes Remote Procedure Calls (NRPC) 156
- Notes Server Dump (NSD) 17
- notes.ini 185
- notes.ini parameter 435
- Notes_AS400_CONSOLE_ENTRIES 200, 224
- Notes_SHARED_POOLSIZ 224
- NotesBench 14, 22, 394
- NotesBench Consortium (URL) 480
- NotesBench workload 22
- NoteSizr 15
- NotesPump 264, 353
- NPC (Notes Remote Procedure Calls) 156
- NPING tool 75
- NRPC 143, 338
- NRPC (Notes Remote Procedure Calls) 156
- NRPC Monitor 157
- NSCOW.jar 227
- NSD (Notes Server Dump) 17
- nsd_all_os400_(julian date)_(time) 17
- NSF buffer pool 219
- NSF buffer pool size 216
- NSF buffer pool size relation to queues 219
- NSF_BUFFER pool size 211

- NSF_Buffer_POOL_Size 134
- NSF_Buffer_Pool_Size 213
 - behavior if omitted 213
- NSF_Buffer_Pool_Size_MB 213
- NSF_DbCACHE_MAXENTRIES 221
- NSF_DbCache_Maxentries 222, 437
- NSF_HOOKS=QNNDIHK 227
- NSFGetFirstLogToArchive() 387
- NSFGetNextLogToArchive() 387
- Number of active threads 305

O

- Obey database quotas during message delivery 208
- Object.Mailobj.NSF.SharedBy.01 464
- Object.Mailobj.NSF.SharedBy.19 464
- Object.Mailobj.NSF.SharedBy.20 464
- Object.Mailobj.NSF.SharedBy.Greater 464
- observation period 172
- ODBC 269, 270
- ODS (On Disk Structure) 308
- ODS 41 385
- OMSGCNV 430
- On Disk Structure (ODS) 308
- Open event log 94
- Open monitor 94
- Operations Navigator 83, 166, 167, 176
 - work with Domino jobs 230
- OptiConnect for OS/400 280
- OptiConnect hub system 280
- OptiConnect satellite systems 280
- optimizing HTTP cache settings 308
- optimizing HTTP logging for performance 310
- optimizing HTTP threads 305
- organization (O) 5
- organizational unit (OU) 5
- OS/400 statistics 153
- OS/400 work management 176
- OSESCTL 431
- OSESHLR 431
- other pool 68
- outbound performance measurements 321

P

- P0FSYNC 373
- Page fault rate 108
- paging option 168
- panel group
 - new extended version released with Domino but not implemented 202
 - update to improve readability 131
- partitioning 274, 323
 - Domino servers 274
 - separate for the R5 SMTP MTA server 202
- partitions 26
- PartnerWorld for Developers (URL) 480
- passthru server 292
- Passthru_LogLevel 443
- PercentAvailSysResources 213, 214, 437
- PerCentReadsInBuffer 212

- performance adjustment 163
- performance analysis concepts 53
- performance data 83
 - from Domino and iSeries 152
- Performance Explorer 53, 116
 - trace points for Domino 117
- Performance Explorer (PEX) 13, 116
 - observe non-Domino workload 366
- performance level 377
- Performance Management/400 12
- Performance Management/400 (PM/400) 12, 21, 480
- Performance Monitor 12, 53, 83, 105
- Performance Navigator 14
- performance recommendations 179
- performance tests
 - with TxL 381
- Performance Tools 53
- Performance Tools/400 12, 114
- personal agent 235
- PEX (Performance Explorer) 13, 116, 366
- PEX Analyzer tool 121
- PEX definition 120
- PEX trace points 118
- PhoneLog 443
- PING command 74
- planning for Domino for iSeries 5
- platform dependent statistics 234, 237, 245, 246
- Platform view 153
- Platform.LogicalDisk 142
- Platform.LogicalDisk* 141
- Platform.LogicalDisk._Total.1._Total.1.PctTime 464
- Platform.LogicalDisk._Total.1._Total.1.PctTime.avg 464
- Platform.LogicalDisk._Total.1._Total.1.PctTime.max 464
- Platform.LogicalDisk._Total.1._Total.1.PctTime.min 464
- Platform.LogicalDisk._Total.1._Total.1.PctUsed 464
- Platform.LogicalDisk._Total.1.AvgQueueLength 464
- Platform.LogicalDisk._Total.1.AvgQueueLength.avg 464
- Platform.LogicalDisk._Total.1.AvgQueueLength.max 464
- Platform.LogicalDisk._Total.1.AvgQueueLength.min 465
- Platform.LogicalDisk._Total.1.PctTime 465
- Platform.LogicalDisk._Total.1.PctTime.avg 465
- Platform.LogicalDisk._Total.1.PctTime.max 465
- Platform.LogicalDisk._Total.1.PctTime.min 465
- Platform.Memory 142
- Platform.memory* 140, 141
- Platform.Memory.FaultsPerSec 465
- Platform.Memory.FaultsPerSec.avg 465
- Platform.Memory.FaultsPerSec.max 465
- Platform.Memory.FaultsPerSec.min 465
- Platform.Memory.KBFree 465
- Platform.Memory.KBFree.avg 465
- Platform.Memory.KBFree.max 465
- Platform.Memory.KBFree.min 465
- Platform.Memory.KBSize 465
- Platform.Memory.PagesPerSec 465
- Platform.Memory.PagesPerSec.avg 465

- Platform.Memory.PagesPerSec.max 465
- Platform.Memory.PagesPerSec.min 465
- Platform.Memory.Wait->Ineligible 466
- Platform.Memory.WaitToIneligible 466
- Platform.PagingFile._Total.Util 466
- Platform.PagingFile._Total.Util.avg 466
- Platform.PagingFile._Total.Util.max 466
- Platform.PagingFile._Total.Util.min 466
- Platform.Process 142
- Platform.process* 141
- Platform.Process.ADMINP.1.Util 466
- Platform.Process.AMGR.1.Util 466
- Platform.Process.AMGR.2.Util 466
- Platform.Process.AMGR.3.Util 466
- Platform.Process.AMGR.4.Util 466
- Platform.Process.HTTP.1.Util 466
- Platform.Process.nadminp.1.Util 466
- Platform.Process.nadminp.2.Util 466
- Platform.Process.nadminp.3.Util 466
- Platform.Process.nadminp.4.Util 466
- Platform.Process.namgr.1.Util 466
- Platform.Process.namgr.2.Util 466
- Platform.Process.namgr.3.Util 466
- Platform.Process.namgr.4.Util 466
- Platform.Process.nhttp.1.Util 466
- Platform.Process.nhttp.2.Util 466
- Platform.Process.nhttp.3.Util 466
- Platform.Process.nhttp.4.Util 466
- Platform.Process.nreplica.1.Util 466
- Platform.Process.nreplica.2.Util 466
- Platform.Process.nreplica.3.Util 466
- Platform.Process.nreplica.4.Util 467
- Platform.Process.nrouter.1.Util 467
- Platform.Process.nrouter.2.Util 467
- Platform.Process.nrouter.3.Util 467
- Platform.Process.nrouter.4.Util 467
- Platform.Process.nserver.1.Util 467
- Platform.Process.REPLICA.1.Util 467
- Platform.Process.REPLICA.2.Util 467
- Platform.Process.REPLICA.3.Util 467
- Platform.Process.REPLICA.4.Util 467
- Platform.Process.ROUTER.1.Util 467
- Platform.Process.SERVER.1.Util 467
- Platform.Process.SERVER.2.Util 467
- Platform.Process.SERVER.3.Util 467
- Platform.Process.SERVER.4.Util 467
- Platform.Process.UPDATE.1.Util 467
- Platform.Process.UPDATE.2.Util 467
- Platform.Process.UPDATE.3.Util 467
- Platform.Process.UPDATE.4.Util 467
- Platform.System 142
- Platform.System.TotalUserUtil 467
- Platform.System.TotalUserUtil.avg 467
- Platform.System.TotalUserUtil.max 468
- Platform.System.TotalUserUtil.min 468
- Platform.System.TotalUtil 140, 468
- Platform.System.TotalUtil.Avg 140
- Platform.System.TotalUtil.avg 468
- Platform.System.TotalUtil.Max 140
- Platform.System.TotalUtil.max 468

- Platform.System.TotalUtil.Min 140
- Platform.System.TotalUtil.min 468
- PLATFORM_STATISTICS_ENABLED 137, 139
- Platform_Statistics_Enabled 443
- platform-dependent statistics 137
- PM/400 12, 88, 102, 480
- POP3 26, 319, 327
 - logging 324
- POP3 (Post Office Protocol Version 3) server 329
- POP3.Command. 468
- POP3.Sessions.Accept.Queue 468
- POP3.Sessions.Active 468
- POP3.Sessions.Inbound.BytesReceived 468
- POP3.Sessions.Inbound.BytesSent 468
- POP3.Sessions.Inbound.Non-SSL 468
- POP3.Sessions.Inbound.SSL 468
- POP3.Sessions.Inbound.SSL.Bad_Handshake 468
- POP3.Sessions.Inbound.Total 468
- POP3.Sessions.Peak 468
- POP3.Sessions.Threads.Busy 468
- POP3.Sessions.Threads.Idle 468
- POP3.Sessions.Total 468
- POP3.UserCache.Attempts 468
- POP3.UserCache.Hits 468
- POP3.UserCache.Inserts 468
- POP3.UserCache.Lookups 468
- POP3.UserCache.OpenReasonCacheFull 468
- POP3.UserCache.OpenReasonDBChanged 468
- POP3.UserCache.OpenReasonNewEntry 468
- POP3_Config_Update_Interval 450
- PORT 199
- port
 - filtering 180
 - local 79
 - mapping 276
 - remote 79
- Previous_TransLog_Path 437
- Previous_TransLog_Status 438
- Previous_TransLog_Style 438
- primary key replication (LEI) 269
- Print Component Report 114
- Print Performance Explorer Report (PRTPEXRPT) 117
- Print PEX Report (PRTPEXRPT) command 116
- Print System Report 114
- Private Memory Pools 169
- process statistics 140
- Processor CIW (Compute Intensive Workload) 349
- processor clock speed 46
- Processor CPW (Commercial Processing Workload) 39, 348
- processor load 67
- processor priority 173
- protected log file 387
- PRPQ
 - 5799-DSD 342
- PRTACTRPT (Print Activity Report) command 115
- PRTCPTTRPT (Print Component Report) command 114
- PRTPEXRPT (Print PEX Report) command 116, 117
- PRTSYSRPT (Print System Report) command 114
- PRTTNSRPT (Print Transaction Report) command 114

- PTF
 - MF27217 345
 - RE01200 344
- pthread_mutex_lock 203
- Public Address Book 11
- public agent 235
- pull replication 249
- push replication 249
- push-pull replication 249

Q

- QACTJOB 80
- QADLACTJ 80
- QADLTOTJ 80
- QAPMJOBL 116
- QASPnn 375
- QBASACTLVL 168
- QBASPOOL 168
- QDYNPTYADJ system value 81
- QDYNPTYSCD system value 81
- QHST 81
- QIBM_QLNT_HTTP_URL 118
- QIBM_QLNT_OPENCOLL 117
- QIBM_QLNT_OPENDB 117
- QIBM_QLNT_OPENDBALL 118
- QIBM_QLNT_SVR_AGENT 118
- QIBM_QLNT_UPDATECOLL 117
- QINTER 169
- QMAXACTLVL 80
- QMCHPOOL system value 81
- QMPGDATA library 14
- qnndiend 224
- QNNDIHK 224, 227
- QNNDISTJ 225, 433
- QNNINADD 224, 227
- QNNINADD task 255
- QNNINAPI 276
- QNNINBRM 179
- QNNINSTS 175, 239, 275
 - Ending 239
- QNOTES 222
- QNOTES (library) 275
- QOS....ResponseTime 469
- QOS.Mail.ISpy..ResponseTime 469
- QOS.Mail.ISpy.ProbeError 469
- QPFRADJ 163, 169, 171
- QPFRADJ system value 81
- QPRCMLTTSK 81
- QSPool 169
- QSYSOPR message queue 171
- QSYSWRK 83, 85, 97
- QTOTJOB 79
- QueryMaxResults 448
- queue access 219
- queue relation to the NSF buffer pool size 219
- queues relate to the NSF 219
- Queuing Multiplier 60, 145
- quotas 205
 - total failure of restore 208
- QUSRNOTES 169, 174, 175, 176

- QUTCOFFSET 97
- QypsChgSysCollectorAttributes 89
- QYPSCSCA 89
- QYPSENDC 89
- QypsEndCollector 89
- QYPSJSVR 97
- QYPSPFRCOL 83, 97
- QYPSSRV 97
- QypsStartCollector 88
- QYPSSTRC 88, 89
- QZDA 350
- QZLSFILE 358
- QZLSSERVER 358

R

- R5 optimized view rebuild 222
- R5Mail
 - performance with TxL 381
 - user 22
 - workload 22
- RAID 175
- RAID0 (mirrored storage protection) 374
- RAID5 (Redundant Array of Independent Disks Type 5) 374
- RDBMS (relational database management system) 261
- RE01200 344
- Real-time monitor 97
- real-time performance data 97
- real-time system performance 90
- record format 262
- Redbooks Web site 481
 - Contact us xviii
- redirecting Domino console command output 128
- REDO 378
- Redundant Array of Independent Disks Type 5 (RAID5) 374
- Referenced Web sites 480
- Refresh Index 190
- Refs 137
- registered users 24, 46
- relational database management system (RDBMS) 261
- reliability 4
- remote access with iNotes 292
- removing the Autostart job entry 225
- REPLICA task 249
- Replica.Cluster 286
- Replica.Cluster.CachedHandles 469
- Replica.Cluster.Docs.Added 469
- Replica.Cluster.Docs.Deleted 469
- Replica.Cluster.Docs.Updated 469
- Replica.Cluster.Failed 469
- Replica.Cluster.Files.Local 469
- Replica.Cluster.Files.Remote 469
- Replica.Cluster.Retry.Skipped 469
- Replica.Cluster.Retry.Waiting 469
- Replica.Cluster.SecondsOnQueue 469
- Replica.Cluster.SecondsOnQueue.Avg 469
- Replica.Cluster.SecondsOnQueue.Max 469
- Replica.Cluster.Servers 469
- Replica.Cluster.SessionBytes.In 469
- Replica.Cluster.SessionBytes.Out 469
- Replica.Cluster.Successful 469
- Replica.Cluster.WorkQueueDepth 469
- Replica.Cluster.WorkQueueDepth.Avg 469
- Replica.Cluster.WorkQueueDepth.Max 469
- Replica.Docs.Added 469
- Replica.Docs.Deleted 469
- Replica.Docs.Updated 469
- Replica.Failed 469
- Replica.Successful 470
- replication backlog 296
- replication topology 6
- REPLICATORS 223
- Replicators 438
- REPORT task 256
- report types 296
- Reporter.Time.Analysis 470
- Reporter.Time.Collectd 470
- Reporter.Time.Elapsed 470
- Request for Comments (RFCs) 317
- reset a statistic 133
- restart server 169
- re-using white space 206
- RFCs 317
- rich text field 260
- ROUTER task 240
- RPC traffic from a Notes workstation 157
- RPG 167
- RSC 344
- RTR_Cached_Handle_Disable 294, 451
- rtr_initial_retry_interval 451
- RTR_Logging 293, 443
- rtr_max_retry_interval 451
- run attributes 64
- Run Domino Command (RUNDOMCMD) command 198
- run priority 64, 66
- RUNDOMCMD (Run Domino Command) command 198
- runpty 174
- runtime performance cache redo 378
- Runtime/Restart Performance 377

S

- SameTime 11
- Sample Domino Server Monitor 105
- SAN (storage area network) 175
- SAVDOMBRM_FILES_IN_GROUP 178, 359
- Save Job Classification Member window 412
- SAVLQPBRM_FILES_IN_GROUP 178
- SBMDOMCMD 128
- SBMDOMCMD (Submit Domino Command) command 198
- scalability 4, 284
- SCHED (Schedule Manager) 252
- scheduling ADMINP 233
- SCM (Supply Chain Management) 263
- scripts 14
- Secondary CPU Utilization 362
- secondary thread 63
- Secondary Workloads 92, 105
- SecureMail 450

- security 5
- SEM.Timeouts 470
- server backups using BRMS/400 179
- server communication 278
- SERVER task 217, 243
- server tasks 223
- Server.<servernameinfo> 470
- Server.Administrator 470
- Server.AvailabilityIndex 470
- Server.AvailabilityThreshold 470
- Server.BootID 470
- Server.BusyTimeQuery.ReceivedCount 470
- Server.BusyTimeQuery.Returned.Objects.Schedules 470
- Server.Cluster 286
- Server.Cluster.AvailabilityIndex 470
- Server.Cluster.AvailabilityThreshold 470
- Server.Cluster.Name 470
- Server.Cluster.OpenRedirects.Failover.Successful 470
- Server.Cluster.OpenRedirects.Failover.Unsuccessful 470
- Server.Cluster.OpenRedirects.FailoverByPath.Successful 470
- Server.Cluster.OpenRedirects.FailoverByPath.Unsuccessful 470
- Server.Cluster.OpenRedirects.LoadBalance.Successful 470
- Server.Cluster.OpenRedirects.LoadBalance.Unsuccessful 470
- Server.Cluster.OpenRedirects.LoadBalanceByPath.Successful 470
- Server.Cluster.OpenRedirects.LoadBalanceByPath.Unsuccessful 470
- Server.Cluster.OpenRequest.ClusterBusy 470
- Server.Cluster.OpenRequest.DatabaseOutOfService 470
- Server.Cluster.OpenRequest.LoadBalanced 471
- Server.Cluster.Portname 471
- Server.Cluster.ProbeCount 471
- Server.Cluster.ProbeError 471
- Server.Cluster.ProbeTimeout(mins) 471
- Server.Cluster.Trans.IntervalAvgTime 471
- Server.Cluster.Trans.IntervalInMinutes 471
- Server.Cluster.Trans.IntervalInSeconds 471
- Server.Cluster.Trans.IntervalsUseInAvg 471
- Server.Cluster.Trans.LastIntervalAvgTime 471
- Server.Cluster.Trans.NormalizeValue 471
- Server.Cluster.Trans.RunningAvgTime 471
- Server.Cluster.Trans.RunningCount 471
- Server.Cluster.Trans.RunningTime 471
- Server.Coprocessor 471
- Server.CPU.Count 471
- Server.CPU.Type 471
- Server.Description 471
- Server.Load 14
- Server.Location 471
- Server.MinIdleDisconnect 471
- Server.MinIdleDisconnect.Time 471
- Server.Monitor.Start 471
- Server.Name 471
- Server.OpenRequest.Maxusers 471
- Server.OpenRequest.PreV4Clients 471
- Server.OpenRequest.Restricted 471
- Server.OpenRequest.V4Clients 471
- Server.Path.Data 472
- Server.Path.Swap 472
- Server.Ports 472
- Server.Ports.RS232 472
- Server.PoweredBy.Notes 472
- Server.Sessions.Dropped 134, 472
- Server.Task 472
- Server.Tasks 472
- Server.Time.Start 472
- Server.Title 472
- Server.Trans.PerMinute 133, 472
- Server.Trans.PerMinute.Peak 472
- Server.Trans.PerMinute.Peak.Time 472
- Server.Trans.Total 472
- Server.User 472
- Server.Users 133, 472
 - 1MinPeak 134
 - 1MinPeakTime 134
 - 5MinPeak 134
 - 5MinPeakTime 134
 - Peak 134
 - PeakTime 134
- Server.Users* 133
- Server.Users.1MinPeak 472
- Server.Users.1MinPeakTime 472
- Server.Users.5MinPeakPeak 472
- Server.Users.5MinPeakTime 472
- Server.Users.Peak 472
- Server.Users.Peak.Time 472
- Server.Version.MTBF 472
- Server.Version.NLM 472
- Server.Version.Notes 472
- Server.Version.NT 472
- Server.Version.OS 472
- Server.Version.OS2 472
- Server.WorkThreads 472
- Server_Availability_Threshold 218, 293, 452
- Server_Cluster_Default_Port 452
- Server_Cluster_On 452
- Server_Cluster_Probe_Port 452
- Server_Cluster_Probe_Timeout 294, 451
- Server_Max_Concurrent_Trans 216, 438
- SERVER_MAXSESSIONS 223
- Server_MaxSessions 217, 438
- Server_MaxUsers 217, 218, 294, 438
- SERVER_POOL_TASKS 64
- Server_Pool_Tasks 439
- Server_Restricted 218
- Server_Session_Timeout 26, 218, 439
- SERVER_SHOW_PERFORMANCE 223
- Server_Show_Performance 223, 439
- Server_TransInfo_Max 294, 451
- Server_TransInfo_Normalize 294
- Server_transinfo_Normalize 452
- Server_TransInfo_Update_Interval 295, 451
- server-based tools 295

- Servers to monitor 106
- ServerTasks 186, 225, 232, 244, 439
- ServerTasksAt2 187
- ServerTasksAt5 244
- Service Level Agreement (SLA) 236
- servlet 271
- Set Stat 133
- SH PE 223
- sh stat calendar 255
- SH STAT DATABASE 221
- SH STAT database.* parameters 215
- SH STAT SERVER 26
- sh ta 128
- sh tran 143
- shared memory pool 168
- shared pools 213
- SharedMail 450
- sharing options 112
- short wait 69
- short wait extended 69, 70
- show DBS 136
- Show Domino open database 136
- show performance 223
- Show Stat 210
- show stat 132
- show stat agent 235
- Show Stat command 132
- Show Stat Database 210
- show stat database 134
- show stat domino 135
- show stat mail.waiting 242
- show stat platform 137
- show task debug 129
- show tasks 186, 244
- show tasks command 128
- Show Trans command 144
- show transactions 143
- Show_Task_Detail 443
- Simple Mail Users (SMU) 45, 394
- single-level storage 372
- size of the storage pool 172
- sizing pools 172
- skip_fixup 439
- SLA (Service Level Agreement) 236
- SLIC (System Licensed Internal Code) 372
- sload.exe 14
- sload50.nsf 14
- SMBCONNMONITOR 358
- SMBCONNWORKER 358
- SMBREQMONITORPKT 358
- SMBSERVERMAIN 358
- SMBWORKER 358
- SMTP 11, 57, 427
 - logging 324
- SMTP (Simple Mail Transport Program) 253
- SMTP messaging best practices 323
- SMTP MIME in Domino R5 202
- SMTP MTA 317
- SMTP MTA in Domino R5 202
- SMTP MTA structure 317
- SMTP.<name>.Ave 472
- SMTP.<name>.Count 472
- SMTP.<name>.Max 473
- SMTP.<name>.Total 473
- SMTP.Command.<CommandName> 473
- SMTP.MessagesProcessed 473
- SMTP.Sessions.Accept.Queue 473
- SMTP.Sessions.Active 473
- SMTP.Sessions.Inbound.BytesReceived 473
- SMTP.Sessions.Inbound.BytesSent 473
- SMTP.Sessions.Inbound.Non-SSL 473
- SMTP.Sessions.Inbound.SSL 473
- SMTP.Sessions.Inbound.SSL.Bad_Handshake 473
- SMTP.Sessions.Inbound.Total 473
- SMTP.Sessions.Peak 473
- SMTP.Sessions.Threads.Busy 473
- SMTP.Sessions.Threads.Idle 473
- SMTP.Sessions.Total 473
- SMTP/MIME MTA 317
- SMTPLDQ (SMTP local deliver request queue) 432
- SMTPMTA 432, 473
- SMTPMTA.ConvFailures 473
- SMTPMTA.Dead 473
- SMTPMTA.Delivered 473
- SMTPMTA.HighInBound 473
- SMTPMTA.HighOutBound 473
- SMTPMTA.InBoundBytes 473
- SMTPMTA.InBoundSize 473
- SMTPMTA.OutBoundBytes 473
- SMTPMTA.OutBoundSize 473
- SMTPMTA.TotalRouted 473
- SMTPMTA.TransFailures 473
- SMTPMTA.Transferred 473
- SMTPMTA.Waiting 473
- SMTPMTA.WaitingConv 473
- SMTPMTA.WaitingRecipients 473
- SMTPMTA.WaitingTrans 474
- SMTPMTA_IPADDR 326
- SMU 45, 394
- SMU (Simple Mail Users) 394
- SNTP 474
- SNTP.<name>.Min 473
- Sound alarm 95, 110
- source directories to a single superdirectory 200
- SPD OptiConnect 280
- Specify Job Classification Category window 410
- Specify Paging Behaviors window 411
- specify workload growth 424
- SPIN.AvtlterToHit 474
- SPIN.Delays 474
- SPIN.DelaysMsecs 474
- SPIN.FRWSemGetReadExt 474
- SPIN.FRWSemPutReadExt 474
- SPIN.Hits 474
- SPIN.MaxDelayMask 474
- SPIN.MaxDelayMsecs 474
- Spool 171
- SQL
 - index 262
 - view 262

- SST 163
- Start BEST/1 command 404
- Start Domino Server 239
- Start Performance Explorer (STRPEX) command 116
- Start System Service Tools (STRSST) command 82
- Starting Collection Services 88
- Statistic Collector 152
- Statistic Monitor 150
- statistical reports 127
- statistics 145
- Statistics & Events database 137
- Statistics & Events database (EVENST4.NSF) 247
- Statistics and Reporting Database 127, 145
- Statistics Collection document 153
- Statistics Reports database for monitoring iSeries and Domino server performance data 153
- STATLOG task 244
- statrep.nsf 127, 137, 138, 145, 174, 234, 237, 245, 246
- statrep400.ntf 153
- STATS task 256
- Stats.Time.Current 474
- Stats.Time.Start 474
- Steady state 207
- STMF 260
- storage area network (SAN) 175
- storage pool activity levels 170
- storage pools 166
- STRDOMSVR 239
- stream files 373
- STRIDOCOL 121
- StringXlate 202
 - Causing severe performance problems 202
- STRPEX (Start Performance Explorer) command 116
- STRPEX (Start PEX) command 116
- STRPFRMON (Start Performance Monitor) command 83
- STRPFRTRC 114
- STRSST (Start System Service Tools) command 82
- Submit Domino Command 128
- Submit Domino Command (SBMDOMCMD) 198
- subnet mask 77
- subsystem name 275
- summary numeric values 108
- Supply Chain Management (SCM) 263
- Suppression time 198
- system and core applications 11
- system ASP 175
- System Collector Attributes 89
- system level 392
- System Licensed Internal Code (SLIC) 372
- system monitor 13, 90
 - modifying existing 96
- system pool 1 171
- system resource utilization 403
- system statistics 139
- system value
 - QDYNPTYADJ 81
 - QDYNPTYSCD 81
 - QMCHPOOL 81
 - QPFRADJ 81, 163
 - QPRCMLTTSK 81

- Systems and Groups 95, 111

T

- TCP/IP Buffer Size 180
- TCPIP 278
- TCPKEEPALV iSeries parameter 218
- TCP-only for Ethernet 181
- Technology-Independent Machine Interface (TIMI) 372
- Technote #167013 185
- Technovations 15
- tell adminp q 233
- tell amgr status 237
- TELL CALCONN QUIT 255
- TELL EVENT QUIT 248
- tests 14
- thread 62
- Thread count 108
- thread number 129
- thread of control 62
- thread state transitions 69, 172
- threads 62, 63
- thresholds 100
- time interval selection 409
- time slice 174
- time slice parameter 66
- time-dependent function 190
- time-dependent selection formulas 189
- timestamp replication (LEI) 269
- TIMI (Technology-Independent Machine Interface) 372
- tips and techniques for designing Domino applications 259
- Tivoli 16
- Tivoli Application Response Measurement (ARM) 16
- Tivoli Manager 16
- Tivoli Manager for Domino 16
- tools 293
- TPROF 123
- TPROF PEX trace 123
- trace points for Domino 117
- Trace Profile 123
- Trace Route (TRACERT) command 75
- TRACERT 75
- TRACERT (Trace Route) command 75
- transaction logging
 - how it technically works 386
 - other benefits 389
- transaction logging (TxL) 371
- transaction logging file 175
- transaction logging mail 205
- Transaction rate 108
- Transaction Rate (Average) 93
- Transaction Rate (Interactive) 93
- Transaction time 108
- transactional log 376
- transactions in Domino 56
- transactions of workload functions 416
- transactions per function 420
- translations (excessive) on iSeries and zSeries 202
- TRANSLOG_AutoFixup 380
- Translog_AutoFixup 440

- TRANSLOG_MaxSize 380
- Translog_MaxSize 218, 440
- TRANSLOG_Path 380
- Translog_Path 440
- TRANSLOG_Performance 380
- Translog_Performance 441
- TransLog_ReCreate_LogCTRL 441
- TRANSLOG_Status 379
- Translog_Status 441
- TRANSLOG_Style 380
- Translog_Style 441
- TRANSLOG_UseAll 380
- Translog_UseAll 218, 442
- triggered 100
- tuning 66
- tuning iSeries performance for Domino 161, 183
- tuning the iSeries server 162
- tuning thread activity 307
- TxL 371
 - performance tests 381
 - related documents 389
- TxL log 387
- TxL log file
 - 64 MB exposure between a closed an open file 388
- typical mail users 28
- typical mail workload 394
- typical user 22, 46, 394

U

- UBM (Unified Buffer Manager) 377, 388
- UDFS (user-defined file system) 374
- UDFS, creating and mounting with CL commands 376
- UNDO 378
- UNDO and REDO records in TxL 387
- UNDO and REDO with recovery as a series of data slices! 387
- UNID (unique ID) 263, 314
- Unified Buffer Manager (UBM) 377
- Uniform Resource Locator (URL) 304
- unique ID (UNID) 263, 314
- UPDALL 185, 187, 191, 192, 193, 198, 257, 389
- UPDATE 173, 185, 188, 190, 191, 244
- Update 185
- UPDATE task 244
- Update_No_Fulltext 220, 448
- Update_Suppression_Limit 448
- Update_Suppression_Time 198, 448
- UPDATERS 223
- Updaters 244, 449
- URL interface 304
- user applications 397
- user ASP 175
- user mail failover 291
- User Pool Faults (Average) 93
- User Pool Faults (Maximum) 93
- user-defined file system (UDFS) 374
- users per Domino server 201

V

- V5R1 391, 480
- validating the BEST/1 model 413
- VFYTCPCNN 78
- View Properties 187
- view properties 190
- View_Rebuild_Dir 334, 442
- views and indices not TxL logged 389
- views, locked 188
- virtual storage 372

W

- wait
 - long 70
 - short 69
 - short extended 70
- Wait-to-Ineligible transitions 172
- WCBT 79
- Web application categorization 313
- Web serving 397
- Web site design 310
- Web.Retriever.Access.FTP 474
- Web.Retriever.Access.Gopher 474
- Web.Retriever.Access.HTTP 474
- Web.Retriever.Bytes.Received 474
- Web.Retriever.Bytes.Sent 474
- Web.Retriever.ImageCache.Hits 474
- Web.Retriever.ImageCache.Misses 474
- Web.Retriever.LogLevel 474
- Web.Retriever.LogMessages 474
- Web.Retriever.Process.Access.FTP 474
- Web.Retriever.Process.Access.Gopher 474
- Web.Retriever.Process.Access.HTTP 474
- Web.Retriever.Process.Bytes.Received 474
- Web.Retriever.Process.Bytes.Sent 474
- Web.Retriever.Process.Num.Active 474
- Web.Retriever.Process.Num.Busy 474
- Web.Retriever.Process.Num.Idle 474
- Web.Retriever.Process.Num.Maximum 474
- Web.Retriever.Process.ProcessID 474
- Web.Retriever.Process.State 474
- Web.Retriever.Process.URLs.Failed 475
- Web.Retriever.Process.URLs.Requested 475
- Web.Retriever.Process.URLs.Succeeded 475
- Web.Retriever.Time.Current 475
- Web.Retriever.Time.Duration 475
- Web.Retriever.Time.Start 475
- Web.Retriever.URLs.Failed 475
- Web.Retriever.URLs.Requested 475
- Web.Retriever.URLs.Succeeded 475
- Web.Retriever.Version 475
- Web.Retriever.VPOOL.Max.Buf 475
- Web.Retriever.VPOOL.Max.Element 475
- Web.Retriever.VPOOL.Max.Marker 475
- Web.Retriever.VPOOL.Max.Text 475
- Web.Retriever.VPOOL.Max.URL 475
- WebAdmin.NSF 313
- WebMail 26, 339
- WebSizr 15

- WebSphere 227
- WebSphere Application Server 343
 - on a DSD 356
- WebSphere Commerce Suite on a DSD 356
- white space 205
- wide area network (WAN) 179
- WLE (IBM Workload Estimator for iSeries) 20
- Work with Active Jobs 176, 186, 192
- Work with Active Jobs (WRKACTJOB) command 177
- Work with BEST/1 Models option 407
- Work with Domino Console (WRKDOMCSL) 277
- Work with Domino Servers (WRKDOMSVR) 277
- Work with Domino Servers (WRKDOMSVR) command 128
- Work with Results display 418
- Work with Shared Storage Pools (WRKSHRPOOL) command 168, 170
- Work with System Activity (WRKSYSACT) command 71, 89, 174, 192
- Work with System Status 168
- Work with System Status (WRKSYSSTS) command 68, 170
- Work with System Status (WRKSYSSTS) display 168
- Work with System Values (WRKSYSVAL) command 79
- Work with TCP/IP Network Status (NETSTAT) command 75
- Work with TCP/IP Network Status (WRKTCPSTS) 75
- workload 23, 414
- Workload Estimator 19
- Workload Estimator (URL) 480
- workload functions 414
- workload measurement 403
- WRKACTJOB 171, 176, 186, 192
- WRKACTJOB (Work with Active Jobs) command 70, 177
- WRKDOMCLS 128
- WRKDOMCSL 277
- WRKDOMSVR 131, 171, 231, 277
- WRKDOMSVR (Work with Domino Servers) command 128
- WRKDSKSTS (Work with Disk Status) command 73
- WRKSHRPOOL (Work with Shared Storage Pools) command 168, 170
- WRKSYSACT (Work with System Activity) command 71, 89, 174, 192
- WRKSYSSTS 168
- WRKSYSSTS (Work with System Status) command 68, 163, 170
- WRKSYSVAL (Work with System Values) command 79
- WRKSYSVAL QPFRADJ 211
- WRKTCPSTS (Work with TCP/IP Network Status) command 75

X

- X.509 330

Z

- zSeries server, excessive translation 202



Redbooks

Domino for iSeries Sizing and Performance Tuning

(1.0" spine)

0.875" x 1.498"

460 <-> 788 pages



Redbooks

Domino for iSeries Sizing and Performance Tuning

Correctly sizing a server with the Workload Estimator

Tips and techniques for tuning Domino for iSeries

Information on iSeries Dedicated Server for Domino

Explore the methodologies and approaches to assist in providing optimal performance of Lotus Domino for iSeries! This IBM Redbook targets technical professionals who are responsible for, or must advise on, the installation and administration of Lotus Domino servers on IBM *@server* iSeries servers. It can also be used by performance specialists to gain knowledge on how to collect, measure, analyze, and extrapolate performance data from Domino servers. This redbook offers you the ability to translate these methodologies into your own situation and come up with specific performance analysis strategies and a set of tuning parameters for the best possible performance.

This redbook shows a detailed approach for:

- ▶ Estimating an appropriate configuration for a new iSeries server to run Domino
- ▶ Measuring Domino application performance
- ▶ Tuning Domino servers and OS/400 resources for optimal use
- ▶ Understanding the impact of many configuration settings
- ▶ Considering partitioning, clustering, the use of text search, indexing, and views
- ▶ Improving throughput of the integration of Domino and DB2 UDB for iSeries
- ▶ Analyzing the impact of the various Domino server tasks
- ▶ Tuning Domino HTTP server workloads

**INTERNATIONAL
TECHNICAL
SUPPORT
ORGANIZATION**

**BUILDING TECHNICAL
INFORMATION BASED ON
PRACTICAL EXPERIENCE**

IBM Redbooks are developed by the IBM International Technical Support Organization. Experts from IBM, Customers and Partners from around the world create timely technical information based on realistic scenarios. Specific recommendations are provided to help you implement IT solutions more effectively in your environment.

For more information:
ibm.com/redbooks