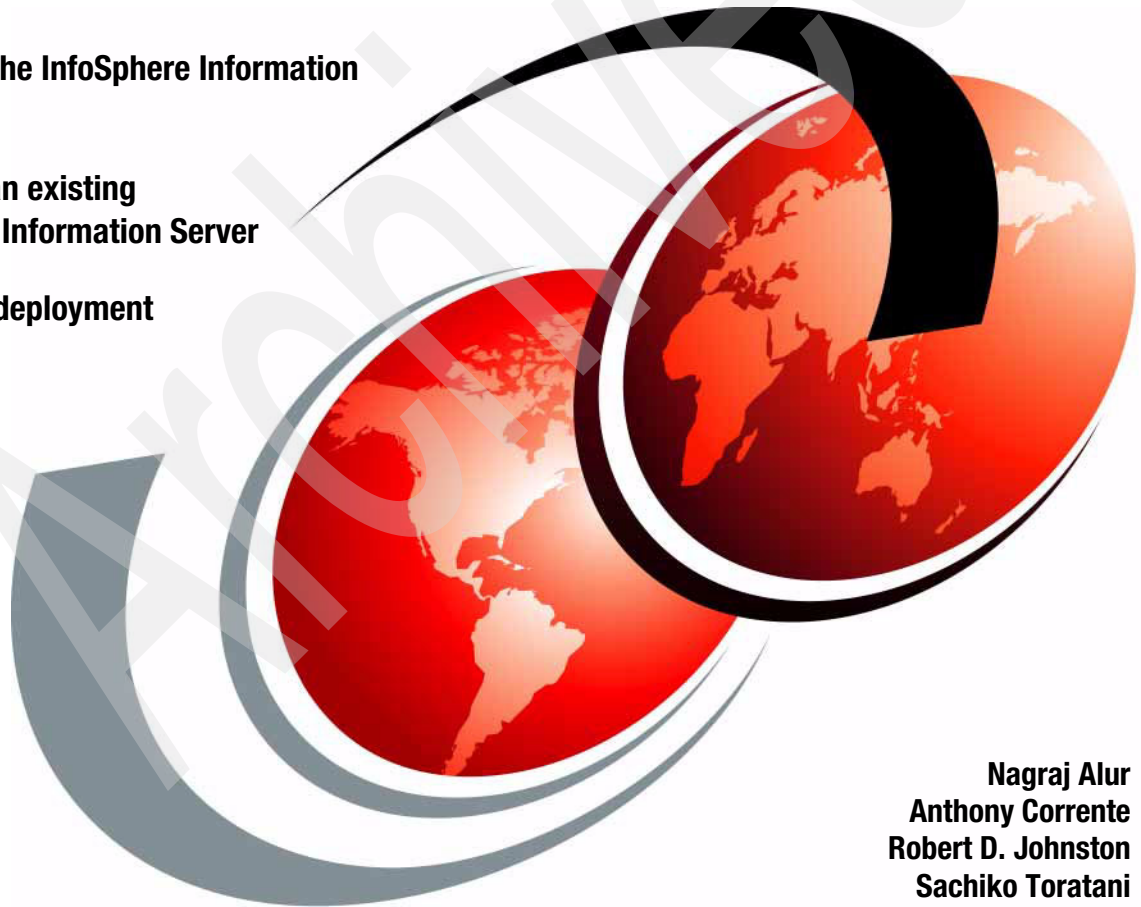


Deploying a Grid Solution with the IBM InfoSphere Information Server

Deploying the InfoSphere Information
Server grid

Migrating an existing
InfoSphere Information Server

Using grid deployment
toolkits



Nagraj Alur
Anthony Corrente
Robert D. Johnston
Sachiko Toratani



International Technical Support Organization

Deploying a Grid Solution with the IBM InfoSphere Information Server

September 2008

Archived

Note: Before using this information and the product it supports, read the information in “Notices” on page xix.

First Edition (September 2008)

This edition applies to Version 8, Release 0, Modification 1 of IBM Information Server (product number 5724-Q36).

© Copyright International Business Machines Corporation 2008. All rights reserved.

Note to U.S. Government Users Restricted Rights -- Use, duplication or disclosure restricted by GSA ADP Schedule Contract with IBM Corp.

Contents

Figures	vii
Tables	xi
Examples	xiii
Notices	xix
Trademarks	xx
Preface	xxi
The team that wrote this book	xxii
Become a published author	xxiii
Comments welcome	xxiv
Chapter 1. IBM InfoSphere Information Server in a grid environment . . .	1
1.1 Introduction	2
1.2 Grid computing overview	2
1.3 Grid computing in an InfoSphere Information Server context	6
1.3.1 Main components	7
1.3.2 Benefits of grid computing	9
1.3.3 Grid definitions	9
1.3.4 Typical grid environments	10
1.3.5 High availability grid environments	15
1.4 InfoSphere Information Server grid computing execution flow	16
1.4.1 Installing Grid Enablement Toolkit	16
1.4.2 How to grid-enable enterprise applications	18
1.4.3 Execution flow	24
1.4.4 Grid computing versus cluster computing	27
1.5 Managing the InfoSphere Information Server grid environment	37
1.5.1 Resource manager	37
1.5.2 Monitoring and tuning	38
1.5.3 Troubleshooting	38
1.6 Successful practices for the InfoSphere Information Server grid environment	39
1.6.1 Configuration guidelines	40
1.6.2 Tuning guidelines	42
1.7 Setting up the InfoSphere Information Server grid-computing environment	43
1.7.1 Step 1: Designing your grid environment infrastructure	43

1.7.2	Step 2: Building and testing your grid infrastructure using the BYOG toolkit	45
1.7.3	Step 3: Reviewing all the jobs	45
1.7.4	Step 4: Installing and configuring the Grid Enablement Toolkit	45
1.7.5	Step 5: Tailoring existing DS/IA jobs to tolerate or exploit the grid environment	46
1.7.6	Step 6: Setting up the management of the grid environment	46
Chapter 2.	Migration scenario	47
2.1	Scenario Overview	48
2.2	Business requirements	48
2.3	Environment configuration	49
2.4	General approach	52
2.5	Step 1: Installing and configuring the grid environment	54
2.5.1	Step 1a: Reviewing existing IBM Information Server environment . .	55
2.5.2	Step 1b: Compiling a grid environment worksheet	63
2.5.3	Step 1c: Installing pre-requisite patches	66
2.5.4	Step 1d: Setting up the Virtual IP and host name and private network	68
2.5.5	Step 1e: Building a Grid environment using the BYOG toolkit	78
2.5.6	Step 1f: Installing and configuring Grid Enablement Toolkit	131
2.5.7	Step 1g: Configuring the standby node	143
2.6	Step 2: Tailoring existing DS/IA jobs to operate in grid environment . .	146
2.6.1	Migrating jobs that require parameter changes only	148
2.6.2	Migrating jobs requiring parameter and stage property changes . .	166
2.7	Step 3: Managing the grid environment	200
2.7.1	Jobs rejected during submission	200
2.7.2	Jobs in the Workload Scheduler LoadLeveler queue	203
2.7.3	Why a job has still not been released from the queue	204
2.7.4	Node status including use of the individual compute nodes	206
2.7.5	Cancelling a waiting job from the queue	214
2.7.6	Adding and removing nodes	217

2.8 Step 4: Failing over to the standby node NILE	221
2.9 Step 5: Failing back to the original front end node ORION	236
Appendix A. Grid deployment toolkits	243
A.1 The toolkits	244
A.2 Build Your Own Grid toolkit	244
A.3 Grid Enablement Toolkit	247
Appendix B. Commands and scripts used in the migration scenario ..	251
B.1 Introduction	252
B.2 Post install scripts	252
B.3 Workload Scheduler LoadLeveler commands	253
B.4 Grid toolkit resource manager commands	254
B.5 Scripts developed	260
B.5.1 Miscellaneous scripts	260
B.5.2 Services scripts	267
B.5.3 HA scripts	289
Appendix C. Parallel Framework	303
C.1 Parallel Framework	304
Appendix D. Additional material	307
D.1 Locating the Web material	307
D.2 Using the Web material	307
D.2.1 System requirements for downloading the Web material	308
D.2.2 How to use the Web material	308
Related publications	309
IBM Redbooks	309
Other publications	309
Online resources	310
How to get Redbooks	310
Help from IBM	311
Index	313

Figures

1-1	Grid computing in InfoSphere Information Server.	8
1-2	NAS grid environment example	11
1-3	SAN grid environment with a single front end node	12
1-4	SAN grid environment with separate NAS server for NFS mounts	13
1-5	Dedicated standby node configuration involving a SAN-configured grid environment	15
1-6	Parameters in the Job Properties window.	18
1-7	Job sequence with sequencer.sh script in the Execute Command Activity stage.	19
1-8	Passing the three values from the sequencer.sh script output	20
1-9	Job sequence with a Execute Command Activity stage for the sequencer.sh script	21
1-10	Execute Command Activity stage for the sequencer.sh script with parameters	21
1-11	Values passed from the sequencer.sh script output to a job in the job sequence	22
1-12	\$APT_GRID_SEQFILE_HOST variable in the Sequential File Properties tab.	23
1-13	Invocation of the Dynamic_Grid.sh script	25
1-14	Dynamic_Grid.sh execution flow.	26
1-15	Cluster configuration, part 1 of 7.	28
1-16	Cluster configuration, part 2 of 7.	28
1-17	Cluster configuration, part 3 of 7.	29
1-18	Cluster configuration, part 4 of 7.	29
1-19	Cluster configuration, part 5 of 7.	30
1-20	Cluster configuration, part 6 of 7.	30
1-21	Cluster configuration, part 7 of 7.	31
1-22	Grid configuration, part 1 of 6	33
1-23	Grid configuration, part 2 of 6	33
1-24	Grid configuration, part 3 of 6	34
1-25	Grid configuration, part 4 of 6	34
1-26	Grid configuration, part 5 of 6	35
1-27	Grid configuration, part 6 of 6	36
1-28	Steps to set up your InfoSphere Information Server grid environment . .	43
2-1	Grid environment configuration.	50
2-2	General approach	52
2-3	Steps in installing and configuring the grid environment.	54
2-4	rowGenSleeper with sequential file, part 1 of 4.	76

2-5 rowGenSleeper with sequential file, part 2 of 4	76
2-6 rowGenSleeper with sequential file, part 3 of 4	77
2-7 rowGenSleeper with sequential file, part 4 of 4	77
2-8 Ganglia	102
2-9 PXE boot execution, part 1 of 6	117
2-10 PXE boot execution, part 2 of 6	118
2-11 PXE boot execution, part 3 of 6	119
2-12 PXE boot execution, part 4 of 6	120
2-13 PXE boot execution, part 5 of 6	121
2-14 PXE boot execution, part 6 of 6	122
2-15 Grid environment variables	140
2-16 Categories of jobs to be migrated to the grid environment	148
2-17 Parameters added to Job Properties for rowGenSleeper_g	148
2-18 Single job rowGenSleeper_g, part 1 of 7	150
2-19 Single job rowGenSleeper_g, part 2 of 7	151
2-20 Single job rowGenSleeper_g, part 3 of 7	151
2-21 Single job rowGenSleeper_g, part 4 of 7	152
2-22 Single job rowGenSleeper_g, part 5 of 7	152
2-23 Single job rowGenSleeper_g, part 6 of 7	153
2-24 Single job rowGenSleeper_g, part 7 of 7	153
2-25 Employee table on which column analysis was performed	160
2-26 Director log output of job BaseProfile_Column_Analysis_Task21780134 (pre-grid)	161
2-27 Information Analyzer job, part 1 of 5	163
2-28 Information Analyzer job, part 2 of 5	163
2-29 Information Analyzer job, part 3 of 5	164
2-30 Information Analyzer job, part 4 of 5	165
2-31 Information Analyzer job, part 5 of 5	166
2-32 Parameters added to Job Properties for rowGenSeqFile_host2_g	168
2-33 \$APT_GRID_SEQFILE_HOST environment variable in Sequential file stage	168
2-34 Sequential file, part 1 of 7	170
2-35 Sequential file, part 2 of 7	171
2-36 Sequential file, part 3 of 7	171
2-37 Sequential file, part 4 of 7	172
2-38 Sequential file, part 5 of 7	172
2-39 Sequential file, part six of 7	173
2-40 Sequential file, part 7 of 7	173
2-41 WebSphereMQConnectorPX stage conductor node pool constraint	176
2-42 \$APT_GRID_SEQFILE_HOST environment variable in Sequential file stage	178
2-43 Parameters added to Job Properties for MQ_g	179
2-44 WebSphere MQ on “conductor” node, part 1 of 11	179

2-45	WebSphere MQ on “conductor” node, part 2 of 11	180
2-46	WebSphere MQ on “conductor” node, part 3 of 11	180
2-47	WebSphere MQ on “conductor” node, part 4 of 11	181
2-48	WebSphere MQ on “conductor” node, part 5 of 11	181
2-49	WebSphere MQ on “conductor” node, part 6 of 11	182
2-50	WebSphere MQ on “conductor” node, part 7 of 11	183
2-51	WebSphere MQ on “conductor” node, part 8 of 11	184
2-52	WebSphere MQ on “conductor” node, part 9 of 11	184
2-53	WebSphere MQ on “conductor” node, part 10 of 11	185
2-54	WebSphere MQ on “conductor” node, part 11 of 11	186
2-55	Multiple jobs rowGen_batch using job sequence	188
2-56	runMultipleJobs_host2_g job sequence	190
2-57	Execute Command Activity Exec Command configuration	191
2-58	rowGenSleeper_g job Job Activity	191
2-59	rowGenSeqFile_host2_g job Job Activity	192
2-60	runMultipleJobs_g Job Properties	192
2-61	Multiple jobs (using Sequencer.sh), part 1 of 11	193
2-62	Multiple jobs (using Sequencer.sh), part 2 of 11	193
2-63	Multiple jobs (using Sequencer.sh), part 3 of 11	194
2-64	Multiple jobs (using Sequencer.sh), part 4 of 11	194
2-65	Multiple jobs (using Sequencer.sh), part 5 of 11	195
2-66	Multiple jobs (using Sequencer.sh), part 6 of 11	195
2-67	Multiple jobs (using Sequencer.sh), part 7 of 11	196
2-68	Multiple jobs (using Sequencer.sh), part 8 of 11	197
2-69	Multiple jobs (using Sequencer.sh), part 9 of 11	198
2-70	Multiple jobs (using Sequencer.sh), part 10 of 11	199
2-71	Multiple jobs (using Sequencer.sh), part 11 of 11	199
2-72	Jobs rejected during submission: Invalid message queue, part 1 of 2	200
2-73	Jobs rejected during submission: Invalid message queue, part 2 of 2	201
2-74	Jobs rejected during submission: Invalid message queue (wrong case), part 1 of 2	201
2-75	Jobs rejected during submission: Invalid message queue (wrong case), part 2 of 2	202
2-76	Jobs rejected during submission: Exceeds maximum nodes threshold, part 1 of 2	202
2-77	Jobs rejected during submission: Exceeds maximum nodes threshold, part 2 of 2	203
2-78	Overview of Data Integration, part 1 of 2	207
2-79	Overview of Data Integration, part 2 of 2	208
2-80	Physical view of nodes at different verbosity levels, part 1 of 2	209
2-81	Physical view of nodes at different verbosity levels, part 2 of 2	210

2-82	Node view of luxor.itsosj.sanjose.ibm.com	211
2-83	Use of the individual compute nodes, part 1 of 3	212
2-84	Use of the individual compute nodes, part 2 of 3	213
2-85	Use of the individual compute nodes, part 3 of 3	214
2-86	rowGenSleeper_g after failover, part 1 of 8	230
2-87	rowGenSleeper_g after failover, part 2 of 8	231
2-88	rowGenSleeper_g after failover, part 3 of 8	231
2-89	rowGenSleeper_g after failover, part 4 of 8	232
2-90	rowGenSleeper_g after failover, part 5 of 8	232
2-91	rowGenSleeper_g after failover, part 6 of 8	233
2-92	rowGenSleeper_g after failover, part 7 of 8	233
2-93	rowGenSleeper_g after failover, part 8 of 8	234
C-1	Process hierarchy of the parallel framework	304
C-2	Runtime control and data networks	306

Tables

- 2-1 Main summary of the existing IBM Information Server environment. . . . 56
- 2-2 Worksheet for grid environment 63
- 2-3 Summary of logs available for review 122
- A-1 Scripts and files provided in the BYOG.tar file 245
- A-2 Scripts and files provided in the grid_enabled.4.0.1.tar file 247

Archived

Examples

2-1	/opt/IBM/InformationServer/ Version.xml file contents	57
2-2	lspci command output	59
2-3	Service network status command output	59
2-4	ifconfig command output	60
2-5	Host Name and gateway address	61
2-6	Nameserver IP address	61
2-7	/etc/hosts file contents	62
2-8	/etc/passwd file contents	62
2-9	/etc/group file contents	62
2-10	/etc/nsswitch.conf file contents	63
2-11	Modify dsenv file	67
2-12	Source the dsenv file as root	67
2-13	Check that no clients or jobs are running	67
2-14	Stop the instance	68
2-15	Start the instance	68
2-16	Verify that the environment variables have taken effect	68
2-17	/etc/sysconfig/network-scripts/ifcfg-eth0 file: Modified contents	69
2-18	/etc/sysconfig/network-scripts/ifcfg-eth0:1 file contents	70
2-19	/etc/sysconfig/network-scripts/ifcfg-eth1 file contents	70
2-20	/etc/sysconfig/network-scripts/ifcfg-eth1:1 file contents	71
2-21	/etc/hosts file: Modified contents	71
2-22	Set the host name in the network file in the /etc/sysconfig directory	72
2-23	Verify the host name	72
2-24	ifconfig command output	72
2-25	ping orion	73
2-26	ping orionpvt	74
2-27	ping kazanpvt	74
2-28	ping kazan	74
2-29	Modify dsenv file	75
2-30	default.apt file: Modified contents	75
2-31	Check if the NFS package is installed	78
2-32	Install the NFS package	78
2-33	Verify that the NFS package is installed	79
2-34	Start the NFS service and verify	79
2-35	/etc/exports file contents	80
2-36	exportfs -a to update the NFS server	80
2-37	Check if the NIS package is installed	80
2-38	Install the NIS package	81

2-39	Verify that the NIS package is installed.	81
2-40	/etc/sysconfig/network file contents.	81
2-41	/var/yp/ypservers file contents.	81
2-42	/var/yp/Makefile file contents.	82
2-43	Set the ypdomainname.	82
2-44	Restart the portmap service.	82
2-45	Configure and start the yp services.	83
2-46	Configure NIS server as master.	83
2-47	make command to rebuild NIS database.	85
2-48	Schedule cron utility.	85
2-49	Generate ssh key for user dsadm.	85
2-50	Copy into the authorized key file and remove execute permission.	86
2-51	~/ssh/config file contents.	86
2-52	Create tftpboot directory structure.	87
2-53	Verify that /tftpboot can be mounted as a file system.	88
2-54	Redhat4.tar.gz file creation from install image into the /tftpboot directory.	89
2-55	Unpack RedHat Linux Enterprise 4 to /tftpboot directory.	90
2-56	Extract BYOG tar file under /tftpboot directory.	91
2-57	Verify /tftpboot structure.	92
2-58	/tftpboot/BYOG/post-install/postenv file contents.	93
2-59	Check if the DHCP package is installed.	93
2-60	Install the DHCP package.	94
2-61	Verify that the DHCP package is installed.	94
2-62	Backup the existing dhcpd.conf file in the /etc directory.	94
2-63	Copy dhcpd.conf from /tftpboot/BYOG/templates directory to /etc directory.	94
2-64	dhcpd.conf file: Modified contents.	95
2-65	dhcpd file: Modified contents.	95
2-66	Configure DHCP service.	96
2-67	Check if the TFTP package is installed.	96
2-68	Install the TFTP package.	96
2-69	Verify that the TFTP package is installed.	96
2-70	/etc/xinet.d/tftp file: Modified contents.	97
2-71	Configure TFTP service.	97
2-72	Check if the Ganglia packages are installed.	98
2-73	Install the Ganglia packages.	99
2-74	Verify that the Ganglia packages were installed.	99
2-75	/etc/gmond.conf file contents.	100
2-76	Configure and start the Ganglia services.	101
2-77	Create Workload Scheduler LoadLeveler group and user.	103
2-78	Check if the Workload Scheduler LoadLeveler LoadL-full-license RPM package is installed.	103

2-79	Install the Workload Scheduler LoadLeveler package	104
2-80	Verify that the Workload Scheduler LoadLeveler LoadL-full-license RPM package is installed	104
2-81	Create directory for loadl local configuration	104
2-82	Install Workload Scheduler LoadLeveler	104
2-83	Initialize Workload Scheduler LoadLeveler on front end node	105
2-84	/home/loadl/LoadL_config file contents	105
2-85	/home/loadl/LoadL_admin file contents	106
2-86	/home/loadl/LoadL_config.local file contents	106
2-87	Resource /home/loadl/.bash_profile	107
2-88	Configure and start Workload Scheduler LoadLeveler as root	107
2-89	Verify that Workload Scheduler LoadLeveler is running	107
2-90	/home/loadl/LoadL_admin file	108
2-91	Anaconda-ks.cfg file in the /root directory as kickstart_template	109
2-92	/root/kickstart_template file: Modified contents	110
2-93	Verify only one row for a range of IP addresses exists in /etc/dhcpd.conf	112
2-94	Create PXE boot config files using PXE_cfg.sh toolkit script	113
2-95	C0A80167 config file contents	113
2-96	network.103 file	114
2-97	Generate kickstart files using the create_ks.sh script	114
2-98	kickstart103.cfg configuration file	114
2-99	DHCP and TFTP server messages in /var/log/messages, part 1 of 2	123
2-100	DHCP and TFTP server messages in /var/log/messages, part 2 of 2	123
2-101	Post install log (/root/post-install) on the compute node	124
2-102	Loadl_config.local on the compute node	129
2-103	/etc/hosts file	129
2-104	/etc/dhcpd.conf file	130
2-105	Extract Grid Enablement Toolkit	131
2-106	Install the Grid Enablement Toolkit	132
2-107	grid_global_values file content	133
2-108	dsenv file: Modified contents	134
2-109	Grid_config.apr file content	135
2-110	Node table: Original contents	136
2-111	Node table: Modified contents	136
2-112	Source the dsenv file	136
2-113	Execute test.sh script	137
2-114	Sample /home/dsadm/grid_jobdir contents while a job is running	139
2-115	Sample /home/dsadm/grid_jobdir contents after a job has finished	139
2-116	Install ganglia	144
2-117	/etc/exports file	144

2-118	NFS status	145
2-119	Virtual IP number 1 for eth0	146
2-120	Virtual IP number 1 for eth1	146
2-121	DSParams file in /opt/IBM/InformationServer/Server/Projects/IS_Grid directory	147
2-122	rowGenSleeper_g Director detail log	154
2-123	Time file in the /home/dsadm/grid_jobdir/ directory	158
2-124	llmonitor command output while job is running	158
2-125	llstatus output while job is running	159
2-126	llmonitor command output after job has finished	159
2-127	rowGenSeqFile_host2_g job Director detail log	174
2-128	llq command (Workload Scheduler LoadLeveler)	203
2-129	listq command (Grid toolkit resource manager command)	204
2-130	List contents of the Workload Scheduler LoadLeveler queue using the listq command	204
2-131	Getting more details on waiting jobs (output of listq orion.85.0)	205
2-132	Cancel a waiting job, part 1 of 3	215
2-133	Cancel a waiting job, part 2 of 3	215
2-134	Cancel a waiting job, part 3 of 3	215
2-135	DataStage Director log relating to the cancelled job orion.85.0	216
2-136	Phoenix and Tarus only (llstatus and listq)	217
2-137	Add Luxor and drop Phoenix	218
2-138	Tarus and Luxor only (llstatus and listq)	220
2-139	ping kasan.itsosj.sanjose.ibm.com	221
2-140	df command output on one of the compute nodes after front end node failure (hangs)	222
2-141	df command output on one of the compute nodes (tarus)	222
2-142	fdisk -l command on the front end node (orion)	223
2-143	df command on the front end node (orion)	223
2-144	fdisk -l command on the standby node (nile)	224
2-145	failover_Nile.sh script output	226
2-146	nodes.cfg file contents	227
2-147	remount_Nile.sh script output	227
2-148	Workload Scheduler LoadLeveler status output while the job was running on standby node nile after failure	235
2-149	failback_Nile.sh script	236
2-150	failback_Orion.sh script output	237
2-151	remount_Orion.sh script output	238
2-152	llmonitor command output on ORION after failback	241
B-1	NFS script	252
B-2	llstatus command	253
B-3	llq command	253
B-4	listq command	254

B-5	Getting more details on waiting jobs	255
B-6	canjob command (cancel a waiting job)	258
B-7	DataStage Director log relating to the cancelled job orion.85.0	259
B-8	llmonitor.sh script	260
B-9	llmonitor output with no jobs running	261
B-10	llmonitor output with jobs running	262
B-11	nisupdatefiles.sh script	263
B-12	IIServer.sh script	264
B-13	runcmd.sh script	267
B-14	dstage.sh script.	268
B-15	ISFagents.sh script.	270
B-16	ISFserver.sh script	274
B-17	IISGrid.sh script	277
B-18	xmeta.sh script	280
B-19	Loadl.sh script.	283
B-20	install.sh script	286
B-21	IIServer output examples	287
B-22	uninstall.sh script	288
B-23	failover_Nile.sh script	289
B-24	mountsanfs_Nile.sh script.	292
B-25	remount_Nile.sh script	293
B-26	failback_Nile.sh script.	294
B-27	umountsanfs_Nile.sh script.	296
B-28	failback_Orion.sh script	297
B-29	mountsanfs_Orion.sh script	300
B-30	remount_Orion.sh script	301

Notices

This information was developed for products and services offered in the U.S.A.

IBM may not offer the products, services, or features discussed in this document in other countries. Consult your local IBM representative for information on the products and services currently available in your area. Any reference to an IBM product, program, or service is not intended to state or imply that only that IBM product, program, or service may be used. Any functionally equivalent product, program, or service that does not infringe any IBM intellectual property right may be used instead. However, it is the user's responsibility to evaluate and verify the operation of any non-IBM product, program, or service.

IBM may have patents or pending patent applications covering subject matter described 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:
IBM Director of Licensing, IBM Corporation, North Castle Drive, Armonk, NY 10504-1785 U.S.A.

The following paragraph does not apply to the United Kingdom or any other country where such provisions are inconsistent with local law: INTERNATIONAL BUSINESS MACHINES CORPORATION PROVIDES THIS PUBLICATION "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF NON-INFRINGEMENT, MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. Some states do not allow disclaimer of express or implied warranties in certain transactions, therefore, this statement may not apply to you.

This information could include technical inaccuracies or typographical errors. Changes are periodically made to the information herein; these changes will be incorporated in new editions of the publication. IBM may make improvements and/or changes in the product(s) and/or the program(s) described in this publication at any time without notice.

Any references in this information to non-IBM Web sites are provided for convenience only and do not in any manner serve as an endorsement of those Web sites. The materials at those Web sites are not part of the materials for this IBM product and use of those Web sites is at your own risk.

IBM may use or distribute any of the information you supply in any way it believes appropriate without incurring any obligation to you.

Information concerning non-IBM products was obtained from the suppliers of those products, their published announcements or other publicly available sources. IBM has not tested those products and cannot confirm the accuracy of performance, compatibility or any other claims related to non-IBM products. Questions on the capabilities of non-IBM products should be addressed to the suppliers of those products.

This information contains examples of data and reports used in daily business operations. To illustrate them as completely as possible, the examples include the names of individuals, companies, brands, and products. All of these names are fictitious and any similarity to the names and addresses used by an actual business enterprise is entirely coincidental.

COPYRIGHT LICENSE:

This information contains sample application programs in source language, which illustrate programming techniques on various operating platforms. You may copy, modify, and distribute these sample programs in any form without payment to IBM, for the purposes of developing, using, marketing or distributing application programs conforming to the application programming interface for the operating platform for which the sample programs are written. These examples have not been thoroughly tested under all conditions. IBM, therefore, cannot guarantee or imply reliability, serviceability, or function of these programs.

Trademarks

IBM, the IBM logo, and ibm.com are trademarks or registered trademarks of International Business Machines Corporation in the United States, other countries, or both. These and other IBM trademarked terms are marked on their first occurrence in this information with the appropriate symbol (® or ™), indicating US registered or common law trademarks owned by IBM at the time this information was published. Such trademarks may also be registered or common law trademarks in other countries. A current list of IBM trademarks is available on the Web at <http://www.ibm.com/legal/copytrade.shtml>

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

AIX 5L™

AIX®

DataStage®

DB2 Universal Database™


DB2®

IBM®

InfoSphere™

LoadLeveler®

Redbooks®

Redbooks (logo) ®

System x™

Tivoli®

UniData®

UniVerse®

WebSphere®

The following terms are trademarks of other companies:

SUSE, the Novell logo, and the N logo are registered trademarks of Novell, Inc. in the United States and other countries.

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

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

Intel, Intel logo, Intel Inside logo, and Intel Centrino logo are trademarks or registered trademarks of Intel Corporation or its subsidiaries in the United States, other countries, or both.

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

Linux is a trademark of Linus Torvalds in the United States, other countries, or both.

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

Preface

This IBM® Redbooks® publication documents the migration of an IBM InfoSphere™ Information Server Version 8.0.1 parallel framework implementation on a Red Hat Enterprise Linux® (RHEL) AS 4 platform to a high availability (HA) grid environment. The grid environment involves five machines that are comprised of one front end node, three compute nodes, and one dedicated standby node that takes over the front-end role when the original front end node fails. Also described are the steps involved in migrating the existing infrastructure to the high availability grid infrastructure and enabling existing IBM InfoSphere DataStage® and IBM InfoSphere Information Analyzer jobs to exploit or tolerate the grid environment.

Note: Other terms used instead of front end include *head* or *conductor*. These terms are defined in 1.3.3, “Grid definitions” on page 9.

This book is aimed at IT architects, Information Management specialists, and Information Integration specialists responsible for delivering cost effective IBM InfoSphere DataStage performance on a RHEL 4.0 platform. This book is organized as follows:

- ▶ Chapter 1, “IBM InfoSphere Information Server in a grid environment” on page 1 provides an overview of grid computing in an InfoSphere Information Server context, and describes its architecture and execution flow. It describes setting up such an environment, discusses the considerations in managing such an environment, and offers best practices suggestions.
- ▶ Chapter 2, “Migration scenario” on page 47 describes an approach to migrating an existing single server InfoSphere Information Server environment on a RHEL Advanced Server 4 platform to a grid environment. The grid environment involves one front end node, three compute nodes, and a dedicated node designated as the “standby” front end node for high availability.
- ▶ Appendix A, “Grid deployment toolkits” on page 243 provides an overview of two toolkits:
 - Build-your-own grid (BYOG) toolkit
This toolkit sets up and configures a Red Hat or SUSE® Linux Grid environment infrastructure.
 - Grid enablement Toolkit
This toolkit configures a Red Hat or SUSE Linux Grid environment.

- ▶ Appendix B, “Commands and scripts used in the migration scenario” on page 251 documents some of the code and scripts used in the migration scenario.
- ▶ Appendix C, “Parallel Framework” on page 303 provides an overview of the Parallel Framework.

The team that wrote this book

This book was produced by a team of specialists from around the world working at the International Technical Support Organization, San Jose Center.

Nagraj Alur is a Project Leader with the IBM ITSO, San Jose Center. He has more than 33 years of experience in database management systems (DBMS), and has been a programmer, systems analyst, project leader, independent consultant, and researcher. His areas of expertise include DBMSs, data warehousing, distributed systems management, database performance, information integration, and client/server and Internet computing. He has written extensively on these subjects and has taught classes and presented at conferences all around the world. Before joining the ITSO in November 2001, he was on a two-year assignment from the Software Group to the IBM Almaden Research Center, where he worked on Data Links solutions and an eSourcing prototype. He holds a master's degree in computer science from the Indian Institute of Technology (IIT), Mumbai, India.

Anthony Corrente is a Senior Technical Support Engineer in Australia. He has 14 years of experience in the database and software technical support field. His areas of expertise include problem determination, down systems support, systems scripting, heterogeneous systems interoperability, with extensive skills in Information Server, DataStage, DB2® for Linux, UNIX®, and Microsoft® Windows®, as well as UniData®, and UniVerse®. He has written a white paper on “Setting up MSCS for DB2 using VMWare.” He holds a degree in Computer Science and Pure Mathematics from University of Sydney.

Robert D. Johnston directs the High Availability (HA) and Grid solutions within the IBM Information Platforms Solutions group. He has more than 20 years experience in DBMS, and has been a Programmer, Systems Analyst, and Project Manager. He has presented at many conferences world wide and provided training on Parallel processing, Grid and HA solutions. Prior to working with Grid solutions, he designed and developed many of the ERP connectivity stages within the DataStage product suite.

Sachiko Toratani is an IT Specialist providing technical support on IBM Information Platforms products to customers in Japan. She has more than 8 years experience in DBMS, and application development in government-related systems. Her areas of expertise include Information Integration and DBMSs, with extensive skills in InfoSphere Information Server, DataStage, and DB2 for Linux, UNIX, and Windows. She is IBM Certified in Database Administrator DB2 Universal Database™ for Linux, UNIX, and Windows. She also contributed to the development of the Redbooks publication *IBM InfoSphere DataStage Data Flow and Job Design*, SG24-7576.

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

John Whyman
IBM Australia

Tony Curcio
Tim Davis
IBM Westboro

Become a published author

Join us for a two- to six-week residency program! Help write a book dealing with specific products or solutions, while getting hands-on experience with leading-edge technologies. You will have the opportunity to team with IBM technical professionals, Business Partners, and Clients.

Your efforts will help increase product acceptance and customer satisfaction. As a bonus, you will develop a network of contacts in IBM development labs, and increase your productivity and marketability.

Find out more about the residency program, browse the residency index, and apply online at:

ibm.com/redbooks/residencies.html

Comments welcome

Your comments are important to us!

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

- ▶ Use the online **Contact us** review Redbooks form found at:

ibm.com/redbooks

- ▶ Send your comments in an e-mail to:

redbooks@us.ibm.com

- ▶ Mail your comments to:

IBM Corporation, International Technical Support Organization
Dept. HYTD Mail Station P099
2455 South Road
Poughkeepsie, NY 12601-5400

IBM InfoSphere Information Server in a grid environment

In this chapter we provide an overview of grid computing in an InfoSphere Information Server context, and describe its architecture and execution flow. We also describe setting up such an environment, discusses the considerations in managing such an environment, and offers successful practices suggestions.

The following topics are covered:

- ▶ Grid computing overview
- ▶ Grid computing in an InfoSphere Information Server context
- ▶ InfoSphere Information Server grid computing execution flow
- ▶ Managing the InfoSphere Information Server grid environment
- ▶ Successful practices for the InfoSphere Information Server grid environment
- ▶ Setting up the InfoSphere Information Server grid computing environment

1.1 Introduction

Although grid computing is not new, applying grid technology to business IT infrastructures represents an exciting evolution of the virtualization, integration, and support of industry standards that IBM has long been applying to solve customers' business problems.

In this IBM Redbooks publication, we describe how the inherent Parallel Framework of InfoSphere Information Server can be extended to leverage grid computing processing. This is achieved through the development of a toolkit that integrates with a resource manager (such as IBM Tivoli® Load Leveler) to enable the dynamic assignment of available nodes to the Parallel Framework engine.

1.2 Grid computing overview

A simple way to understand grid computing is a scenario where all of the disparate computers and systems in an organization, or among organizations, become one large, integrated computing system. That single system can then be turned loose on problems and processes too large and intensive for any single computer to easily handle alone in an efficient manner.

Under non-grid circumstances the following situations apply:

- ▶ Mainframes lie idle 40% of the time
- ▶ UNIX servers are actually “serving” something less than 10% of the time
- ▶ Most PCs do nothing for 95% of a typical day

At its simplest, a grid can gather untapped power and functionality and offer the following benefits:

- ▶ Make it available to users across the grid: where it is needed, when it is needed
- ▶ Offer a major gain in sheer power and speed
- ▶ Enable seamless file and data sharing from heterogeneous sources, connecting multiple users

The benefits of grid computing go far beyond power and speed. Grid computing also represents an important evolution in sharing and collaboration among a variety of advanced proprietary, open, and homegrown tools and systems. Therefore, a grid also can provide these benefits:

- ▶ Embrace multiple organizations
- ▶ Create a virtual organization (VO) of various departments within and among larger organizations
- ▶ Bring people and computing power together to solve highly complex problems with greater ease and agility than ever before

To each user on the grid, all of this complexity is hidden. Just as an Internet user views a unified instance of content through the Web, a grid user essentially sees a single, large virtual computer. Unleashing and centralizing the latent power of multiple processing, network bandwidth, and storage capacity, grid grants both users and applications seamless access to vast IT capabilities, with a sense of a single view. To bring together such power and ease of use, grid computing uses a set of technologies that represent the latest developments in integration, openness, and virtualization. In fact, grid computing can be seen as the latest and most complete evolution of more familiar developments with which IBM customers are already comfortable:

- ▶ Like the Web, grid computing keeps complexity hidden. Multiple users enjoy a single, unified experience.

Unlike the Web, which mainly enables communication, grid computing enables full collaboration toward common business goals.

- ▶ Like peer-to-peer, grid computing allows users to share files.

Unlike peer-to-peer, grid computing allows many-to-many sharing. This includes other resources as well as files.

- ▶ Like clusters and distributed computing, grids bring computing resources together.

Unlike clusters and distributed computing, which require physical proximity and operating homogeneity, grids can be geographically distributed and heterogeneous.

- ▶ Like virtualization technologies, grid computing enables the virtualization of IT resources.

Unlike virtualization technologies, which virtualize a single system, grid computing enables the virtualization of vast and disparate IT resources.

It was in the mid-1990s that IT professionals working on advanced science and technology projects began using the term “grid” to refer to their large-scale, highly powerful distributed computing deployments. In general, grid computing may be loosely categorized into two basic usage types:

- Scientific grid computing

Scientific grid computing can be defined as scavenging idle server and desktop capacity through the Internet or within the firewall of the enterprise to cost effectively process large-scale computational problems.

In the popular mind, perhaps the best-known grid is an effort by the Search for Extraterrestrial Intelligence (SETI) Institute, where more than 2 million volunteers from all over the world have donated idle processing time on their computers to searching through the signals collected every day by the world's largest radio telescope. Meanwhile, the Human Genome Project has used grid technology to perform some of the most daunting sequencing tasks in cracking the human genetic code.

Commercial scientific grid computing is typically used to sift through large volumes of data, or to calculate large equations. For example, bioscience companies use grids to process genetic algorithms, and oil companies use grids to interpret seismic data from oil explorations.

- Enterprise grid computing

Enterprise grid computing can be defined as using dedicated farms of low-cost, uniformly configured servers to replace large UNIX clusters or mainframes. Most commonly, this is used to handle commercial data processing workloads and manage service level agreements (SLA) at the optimal price/performance level. An example of this is an Enterprise Data Warehouse with large volumes of data to de-duplicate and load.

Enterprise grid computing is typically used to provide high availability and scalability to high volumes of transactions, or to process large volumes of data into analytical information. This type of processing was traditionally run on mainframes, and large-scale UNIX platforms. Typical examples include things like manufacturing product quality analysis and telecommunication customer churn analysis. Likewise, most large Data Service providers have also moved to GRID computing to drive the backbones of their data processing and delivery supply chains.

Regardless of whether grids are used to support compute-intensive applications in research or academia or data-intensive applications in enterprise data centers, they share the following important characteristics:

- ▶ Commodity hardware and operating systems

Grids leverage lower-cost hardware and operating systems to complete their processing, rather than traditional higher-cost SMP platforms. They may achieve a price/performance advantage by doing this.

- ▶ Anonymous parallel execution

Applications are unaware of the computers in the grid that are executing a given task and the number of computers assigned to execute the task.

- ▶ Utility usage model

Grids can be expanded simply by adding nodes and servers to a centralized grid server farm instead of purchasing dedicated departmental SMP application servers. This can lower the total cost of ownership (TCO) and enables adherence to SLA obligations. Ultimately, the cost of using the capacity on the grid may be recouped by charging a department or customer for the cycles required to run their application portfolio over a set period.

The centralized shared services grid approach to application deployment is in marked contrast to the traditional practice of running an important application on a dedicated SMP computer purchased to run a single application for an individual department. For example, consider how business applications such as ERP systems are managed today. When a single server fails, pagers beep and people scramble to resolve the problem quickly.

By contrast, for the grid innovators, their departments have stopped purchasing dedicated systems for individual applications and instead have multiple applications running on their central corporate grid infrastructure. When a failure occurs in a grid node, other nodes automatically pick up the load without administrative intervention, resulting in no downtime (or limited application recovery downtime) for the application user. Not only does the grid routinely provide greater availability and service levels, but it does so at a lower TCO. This is due in part to the price/performance advantages of hardware that can replace failed components inexpensively, and add capacity.

For more information about IBM Grid Computing, refer to the following Web page:

<http://www-03.ibm.com/grid/>

1.3 Grid computing in an InfoSphere Information Server context

The Parallel Framework engine of InfoSphere Information Server enables IBM InfoSphere DataStage, QualityStage, and Information Analyzer jobs to run in parallel on a single SMP server, or on multiple servers in a clustered environment. In both cases, the degree of parallelism used by a job is determined by the nodes (and correspondingly the servers/machines) specified in the configuration file associated with it. To change the degree of parallelism, or the servers on which a job runs, you must modify the configuration file with the new number of nodes and its associated servers. This makes the association of nodes used by a job static.

With a grid implementation, the static configuration files normally defined for a cluster are replaced with dynamic configuration files created at runtime. This is accomplished with the Grid Enablement Toolkit.

Section 1.4.4, “Grid computing versus cluster computing” on page 27 provides a detailed explanation of the differences between a cluster and grid implementation using the same job submission.

Note: Although grid computing has been defined elaborately in 1.2, “Grid computing overview” on page 2, its definition in an InfoSphere Information Server context refers to the use of many low-cost commodity servers with high speed CPU's to service compute-intensive operations where any node can service the request. The restriction is that each server in the grid must be running the same operating system and on the same backbone.

The following sections briefly describe the following topics:

- ▶ Main components
- ▶ Benefits of grid computing
- ▶ Grid definitions
- ▶ Typical grid environments
- ▶ High availability grid environments

1.3.1 Main components

Figure 1-1 on page 8 shows the main components of an InfoSphere Information Server grid implementation.

Note: For a brief explanation of terms used in Figure 1, refer to 1.3.3, “Grid definitions” on page 9.

The key components are the Grid Enablement Toolkit and the Grid Resource Manager whose main functions are as follows:

- ▶ Grid Enablement Toolkit is a set of scripts and templates used to create a dynamic configuration file (\$APT_CONFIG_FILE) based on using a resource manager that identifies idle servers. Its main functions are as follows:
 - Coordinate the activities between the Parallel Framework and the resource manager.
 - Create the parallel configuration file from a template.
 - Log the activity.

This toolkit is described briefly in “Grid Enablement Toolkit” on page 247.

- ▶ Grid Resource Manager identifies idle servers in a pool of available servers. The following resource managers are supported:
 - IBM LoadLeveler®
 - SGE
 - Platform LSF
 - PBSpro/Torque/OpenPBS
 - Condor
 - DataSynapse GridServer

Note: The pool of available servers can be expanded or contracted as requirements change, and the new pool will be automatically consumed by the jobs.

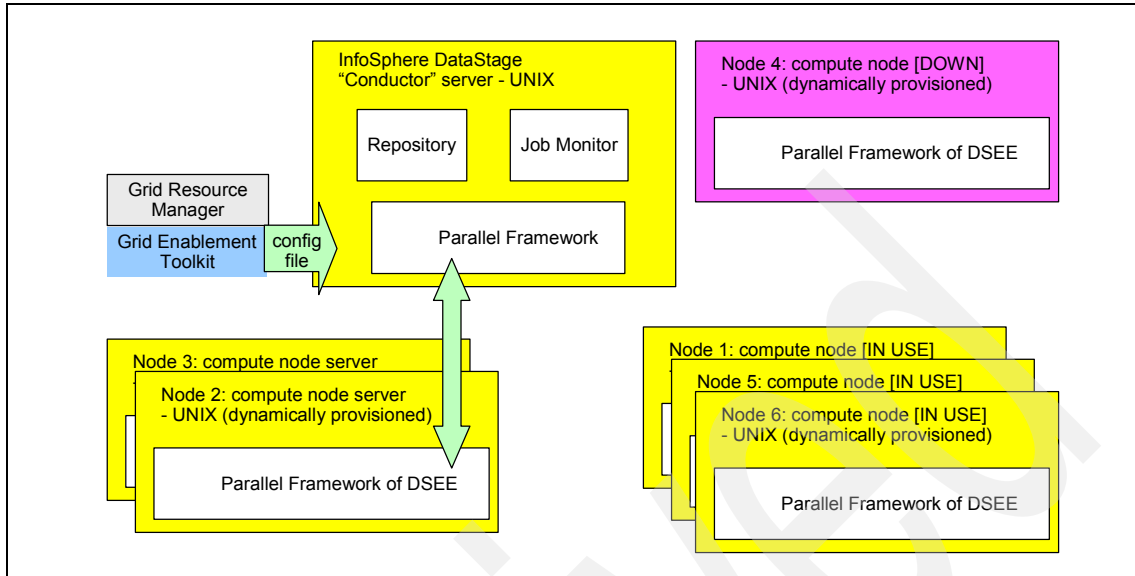


Figure 1-1 Grid computing in InfoSphere Information Server

Briefly, the grid implementation works as described in the following process. For more detail, see 1.4, “InfoSphere Information Server grid computing execution flow” on page 16.

1. When a job is submitted, it is intercepted by a component of the Grid Enablement Toolkit. The component generates and submits a script to the resource manager that gets placed in a Grid Resource Manager queue.

Note: The toolkit provides an interface into the resource manager, allowing the current run methods for the InfoSphere Information Server applications to be used with minimal impact.

The script remains in the queue until the resources (appropriate number of nodes and specific nodes if constraints are placed for them in one or more stages in the job) required by the job become available.

2. After the node or nodes requested become available as monitored by the resource manager, the submitted script generates a parallel configuration file based on those specific nodes and starts the job.

While the job is running, the resource manager prevents additional jobs from using the same resources until the running job completes.

3. The submitted script waits until the parallel job finishes before performing a final cleanup and release of assigned resources.

1.3.2 Benefits of grid computing

Grid computing in an InfoSphere Information Server context is an attractive solution to IT departments because it provides the following benefits:

- ▶ Low cost, high availability, and quick turnaround
 - Low cost hardware (two physical CPUs + Dual Core, 4 GB memory requirement)
 - Minimum OS license fees
 - Unavailable servers do not impact the entire IT department because available servers are dynamically assigned and unavailable servers are automatically excluded from consideration
- ▶ Resource Management software determines which servers to use
- ▶ Concurrent job processing
- ▶ Consistent run times provide better SLA with users

1.3.3 Grid definitions

Grid computing in an InfoSphere Information Server context introduces certain terms that are used to clarify configuration and usage within a grid.

Node	Server within the grid
Partition	Framework degree of Parallelism per node. Default is 1.
Frontend, head or conductor node or server	Main server in the grid that provides software (such as DataStage) or services to the compute nodes in the grid. May also be referred to as the DataStage conductor node.
Compute Node	Server that typically does not have permanent data storage devices, but provides pure compute power (CPU) to a submitted job.
System Management	Tools used to manage servers within the grid environment such as NIS (user/passwd) and PXE booting (network boot).
Resource manager	Provides the necessary scheduling and monitoring tools to distribute jobs to nodes that are available within the grid.
Resource Monitor	Software used to monitor hardware use of the grid.

1.3.4 Typical grid environments

Grid environments with InfoSphere Information Server vary depending upon your particular current configuration or IT environment, such as the use of network-attached storage (NAS) or storage area networks (SAN).

Whatever configuration you implement, a server must be designated as the primary server or front-end server, and multiple servers must be designated as compute nodes.

- ▶ All DataStage, QualityStage, and Information Analyzer jobs are initiated from the front-end server.
- ▶ Compute nodes are where the jobs are processed.

The front-end server must be able to accommodate messages from all concurrently submitted jobs. Data is moved between the various nodes based on job partitioning requirements by the Parallel Framework components using TCP/IP (ports $\geq 11,000$). The ports in the 10,000 range are used for conductor-to-section-leader communication. The topology of the network with one gigabit switches must be managed to isolate jobs to a given gigabit switch whenever possible.

In the following subsections, we describe:

- ▶ NAS configuration grid environment
- ▶ SAN configuration grid environments
- ▶ NAS versus SAN configurations
- ▶ Private networks

NAS configuration grid environment

NAS systems can be an inexpensive and easy solution for grid computing. Configuration of NAS devices for the Parallel Framework is simple and presented to the compute nodes as an NFS file system. When using NAS devices in an InfoSphere Information Server environment, read/write I/O occurs on compute nodes using NFS mounts of the NAS device.

Using a NAS device allows for the configuration of multiple head nodes. You might consider this option for the following reasons:

- ▶ You want to provide isolation between projects for security purposes. For example, project A can be isolated from project B.
- ▶ You want to provide a shared services environment where all compute nodes are shared between development, test, and quality assurance and production head nodes.

- Instead of having one large front end node, you can distribute the projects across many multiple smaller nodes.
- You can have different InfoSphere Information Server version and release levels on the different head nodes, which enables development, regression testing and other scenarios to use the same compute nodes.

Figure 1-2 shows a grid environment involving NAS devices with the nodes responsible for all activities related to InfoSphere Information Server applications. Reads and writes are performed using NAS that is exposed as either NFS, Parallel Virtual File System (PVFS) or Global File System (GFS) file systems.

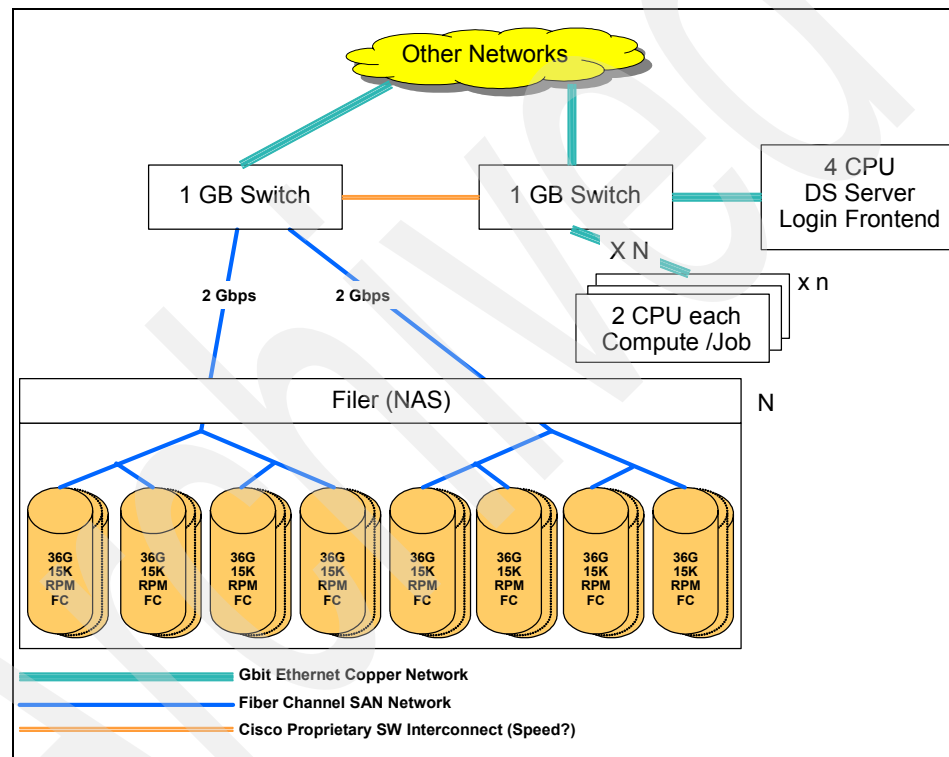


Figure 1-2 NAS grid environment example

SAN configuration grid environments

There are two possible SAN-type configurations that are supported for the Parallel Framework engine of InfoSphere Information Server:

- ▶ The most common configuration (shown in Figure 1-3) is one in which the front end node provides for all file I/O for the grid. This is accomplished by having the front end node provide all NFS mounts to the compute nodes. This involves having a single front end node for the grid and having enough network bandwidth (through a private network if possible as shown in Figure 1-3) from the front end node to the compute nodes for all the I/O activity through the use of multiple Network Interface Cards (NICs).

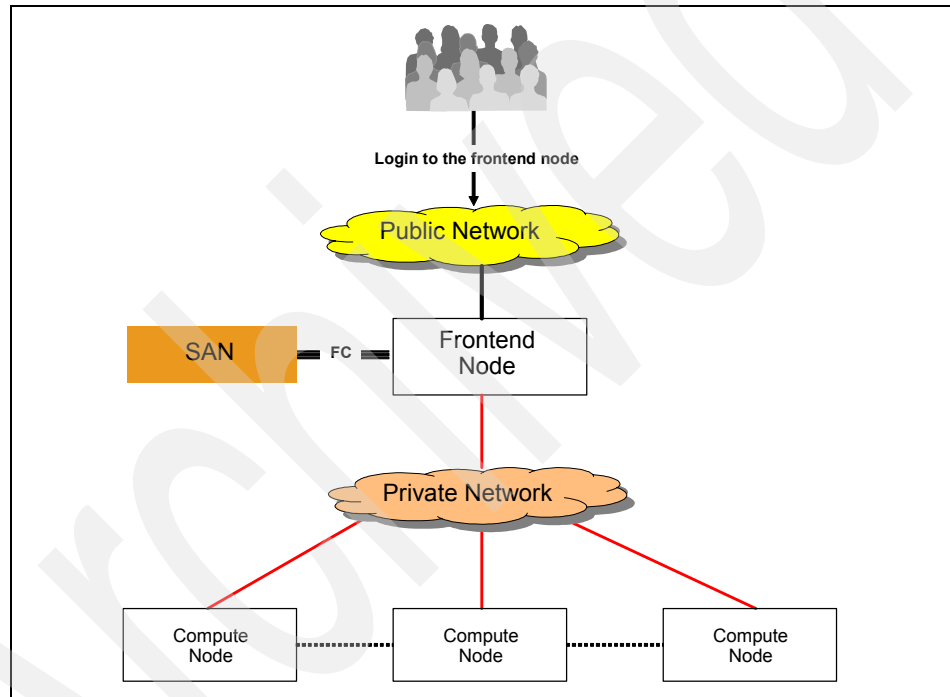


Figure 1-3 SAN grid environment with a single front end node

- ▶ An alternative configuration (shown in Figure 1-4 on page 13) that allows more flexibility by using a separate NAS Server (that has the SAN storage attached) from which NFS mounts occur to the compute nodes. This is basically a simulated NAS device, but built using existing hardware instead of buying new disk storage. This type of configuration allows for expansion by adding additional nodes attached to the SAN and making the file system readable to the compute nodes through NFS. The “resource disk” entry within the APT_CONFIG_FILE can be used to distribute disk storage by NIC card/mount points.

Attention: A SAN configuration can support multiple front end nodes if a Global File System (GFS) is used. GFS is a shared disk file system for Linux clusters. It allows all nodes to have direct concurrent access to the same shared block storage. Without GFS, a SAN configuration can only have one front end node.

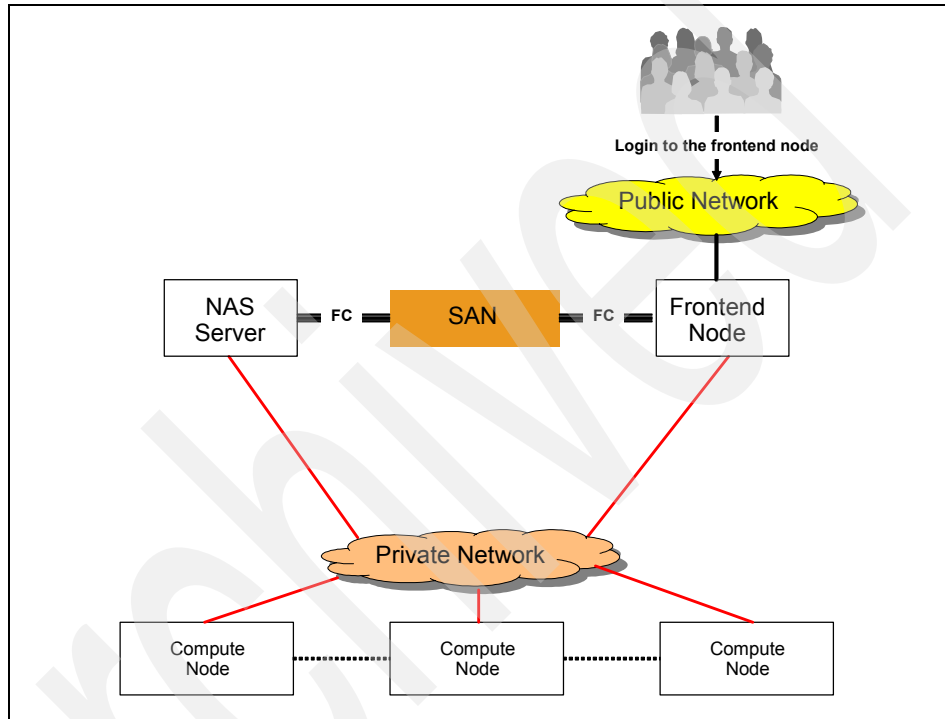


Figure 1-4 SAN grid environment with separate NAS server for NFS mounts

NAS versus SAN configurations

NAS configuration is the recommended solution over SAN for grid implementations for the following reasons:

- ▶ NAS configuration is simple and allows for multiple front end nodes sharing the same compute resources, while a SAN configuration requires something like GFS to support multiple front end nodes.
- ▶ A SAN configuration requires multiple NIC cards in the front end to NFS share the file systems to the compute nodes given the limitation of 1 GB limit of each NIC card.

- ▶ Based on I/O requirements, you might require two or more NIC cards for the NFS mounts.
- ▶ In a SAN configuration with more than one front end node, the compute nodes cannot identify the specific front end node /home directory to use without GFS. With GFS, however, the compute nodes are still only able to share the one /home directory between both front end nodes.
- ▶ With a NAS configuration, there is only one /home directory for each user. Different users can access different /home directories. Software mount points are defined based on a specific front end node, thereby avoiding any confusion about the specific /home directory to be used. As mentioned earlier, you can support different uses such as development and regression testing from the same compute nodes.

Private networks

Large amounts of data and messages are moved between the front end node and the compute nodes, which can place significant bandwidth demands on the gigabit switches. A separate private network is recommended for this data movement through the use of multiple NIC cards for public and private network connections. This is particularly applicable to SAN-configured grid environments described in “SAN configuration grid environments” on page 12.

When compute nodes have multiple NIC cards for public and private network connections, there is a chance that the actual host name of the nodes is not in the private network, but in the public network. Based on the resource manager used, this causes the routing of the activity to occur on the public network rather than the desired private network. To avoid this possibility, you must translate the node name into a private network name using a node translation process which is run whenever the dynamic APT_CONFIG_FILE is created.

The node translations can be performed in the following manners:

- ▶ Front end node can be translated using the APT_PM_CONDUCTOR_HOSTNAME environment variable within the dsenv file as shown in Example 2-29 on page 75.
- ▶ A Node_table file located the \$GRIDHOME directory contains a list of compute node names that must be translated to use a private network when routing data. All nodes other than the front end node must be translated using this method. See Example 2-110 on page 136 and Example 2-111 on page 136.

1.3.5 High availability grid environments

A high availability (HA) grid environment is similar to one with a non-grid environment with the following considerations:

- ▶ A dedicated standby node (as shown in Figure 1-5) or a shared standby node may be used.

In the case of a shared standby node, the workload executing on the shared node may be a non-grid workload, or one of the compute nodes currently executing the grid workload.

In a majority of cases, one of the compute nodes in the grid environment serves as the standby shared node.

- ▶ The standby node requires the following factors to be enforced:
 - HA software runs on the standby compute node and the front end node.
 - CPU, memory, and disk requirements on the standby node must match that of the front end node.

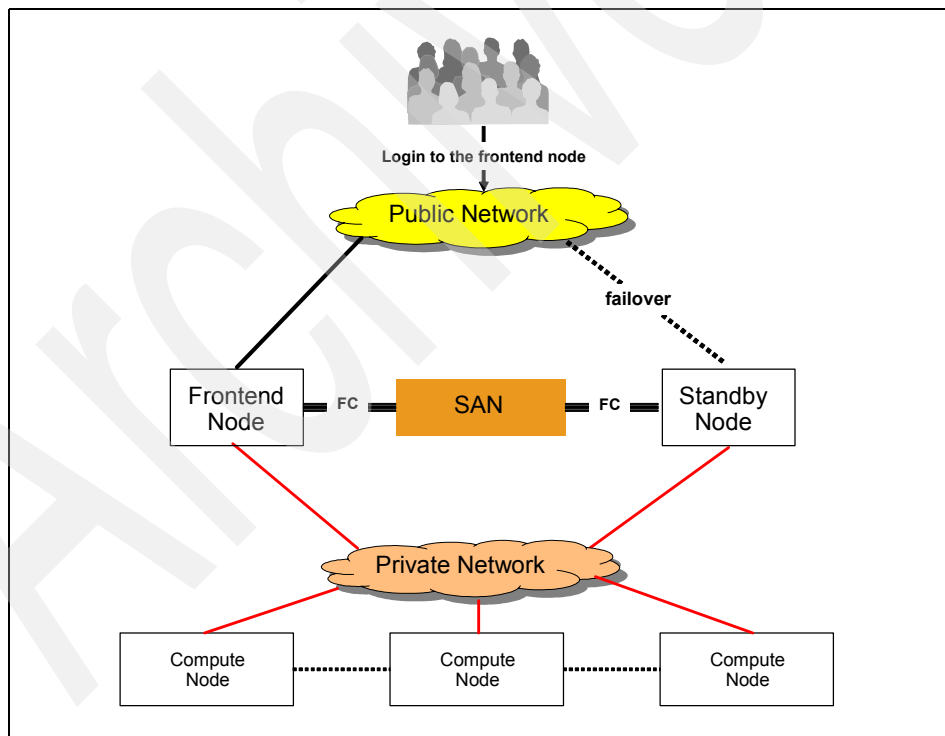


Figure 1-5 Dedicated standby node configuration involving a SAN-configured grid environment

Implementing HA is easier in a NAS-configured grid environment than in a SAN-configured grid environment.

Implementing HA in a SAN-configured grid environment requires the following factors:

- ▶ SAN cards in front end node and the standby node
- ▶ NFS to compute nodes using the virtual IP and not the physical host name of the front end node

1.4 InfoSphere Information Server grid computing execution flow

The main components of grid environment and the role of the Grid Enablement Toolkit and the Grid Resource Manager were described briefly in 1.3.1, “Main components” on page 7.

In this section, we elaborate on some of these roles as well as describe the procedure for grid-enabling a job and job sequence, and highlight the difference between cluster computing and grid computing:

- ▶ Installing Grid Enablement Toolkit
- ▶ How to grid-enable enterprise applications
- ▶ Execution flow
- ▶ Grid computing versus cluster computing

1.4.1 Installing Grid Enablement Toolkit

The Grid Enablement Toolkit installer performs the following tasks:

1. Gathers the following information from the user about their grid environment
 - (Information) Server type
Specifies whether this is an Information Server Install or a (DataStage) 7.x install.
 - Path to resource manager commands
Specifies the resource manager to be used and its location.

- (Work) Directory (full path)

Specifies the grid JOBDIR work directory that must be shared on all nodes and includes the following files:

- Signal files
- Configuration files
- Management scripts
- Create directory permissions 777

Note: You may tighten these permissions to 770 and define owner and group as dsadm and dstage.

- Maximum number of nodes per job (global)
 - List of valid and default queues
 - Whether system defaults should be used for all projects
 - Java™ path
2. Modifies the dsenv file, a centralized file on the DataStage server for storing environment variables. This enables InfoSphere Information Server to contact the resource manager, and to invoke the Java code that creates the configuration file and submits the jobs. To achieve this, it performs the following tasks:
 - Adds \$GRIDHOME
 - Includes \$GRIDHOME and Java path in \$PATH
 - Identifies the location of the resource manager executables/environment
 3. Creates grid environment variables in each project such as APT_GRID_COMPUTENODES, APT_GRID_ENABLE, and APT_GRID_PARTITIONS.
 4. Creates grid configuration values in the grid_global_values global configuration file and the grid_config.apl default configuration file as the template for all jobs.
 5. Enables the intercept for interacting with the resource manager by performing the following tasks:
 - Renaming osh to osh.exe
 - Renaming osh.sh (provided with the Grid Enablement Toolkit) to osh
 - The renamed osh script calls either osh.exe (the original osh executable) or the osh_conductor.sh script.
 - The osh_conductor.sh script (provided with the Grid Enablement Toolkit) is called once per job startup to validate or begin the process of creating a configuration file.

The test.sh script is provided to verify the successful installation of the Grid Enablement Toolkit. Perform the following steps to verify your installation:

1. Source the .dsenv file.
2. Run test.sh as non-root user.
3. Confirm that there are no errors in the output of the script.

1.4.2 How to grid-enable enterprise applications

All DataStage, QualityStage, and Information Analyzer jobs must be modified to some degree to enable them to exploit the grid environment. The procedure for DataStage and QualityStage are identical, while IBM InfoSphere Information Analyzer jobs have a different procedure. These are described here as follows.

DataStage and QualityStage enablement

DataStage and QualityStage single jobs and job sequences can be grid-enabled as described in the following subsections. If a Sequential File stage is present in the job, then the stage properties must be modified.

Single job enablement

To create a dynamic configuration file for a single job, set the following parameters as shown in Figure 1-6:

- ▶ \$APT_GRID_ENABLE = YES
- ▶ \$APT_GRID_COMPUTENODES= 3

This value must be between 1 and MaxNodesPerJob where MaxNodesPerJob is specified in the grid_global_values or DSParams file for each project.

- ▶ \$APT_GRID_PARTITIONS=1

Specifies the number of partitions per compute node. Can have a value of 1 through n.

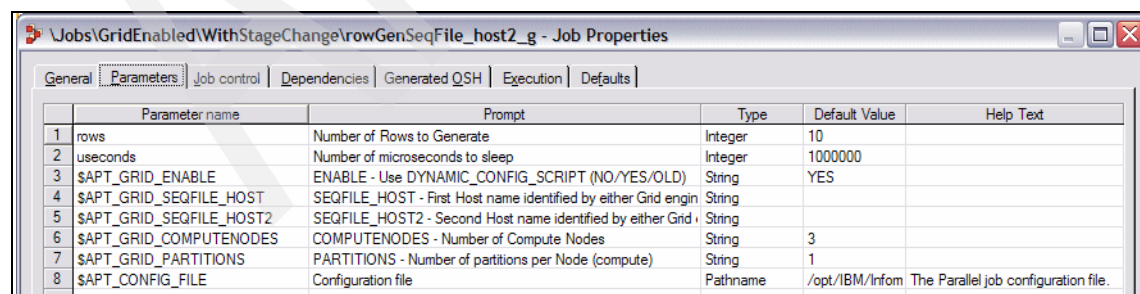


Figure 1-6 Parameters in the Job Properties window

The \$APT_GRID_SEQFILE_HOST and \$APT_GRID_SEQFILE_HOST2 environment variables are only required when Sequential File stages are defined in the job. This is described in “Sequential File processing” on page 22.

Job sequence enablement

A job sequence allows you to specify a sequence of parallel jobs or server jobs to run. The sequence can also contain control information. For example, you can specify different courses of action to take depending on whether a job in the sequence succeeds or fails. After you have defined a job sequence, it can be scheduled and run using the DataStage and QualityStage Director. It appears in the InfoSphere Information Server repository and in the DataStage and QualityStage Director client as a job.

To create a dynamic configuration file for a job sequence, the sequencer.sh script must be invoked, which in turn calls the Dynamic_grid.sh script.

The current sequencer.sh script performs the following tasks:

- ▶ Accepts the following command line attributes as input:
 - -f <name of new apt_config_file>
 - -c <number of compute nodes>
 - -p <number of partitions per node>
- ▶ Produces a single line space-delimited output that has three sets of values that identify APT_GRID_SEQFILE_HOST, APT_GRID_SEQFILE_HOST2 and APT_CONFIG_FILE.

These three values can be accessed in downstream job activity stages using the expression: Field(<stagename>.\$CommandOutput, " ", <fieldnum>)

This is shown in Figure 1-7 and Figure 1-8 on page 20.

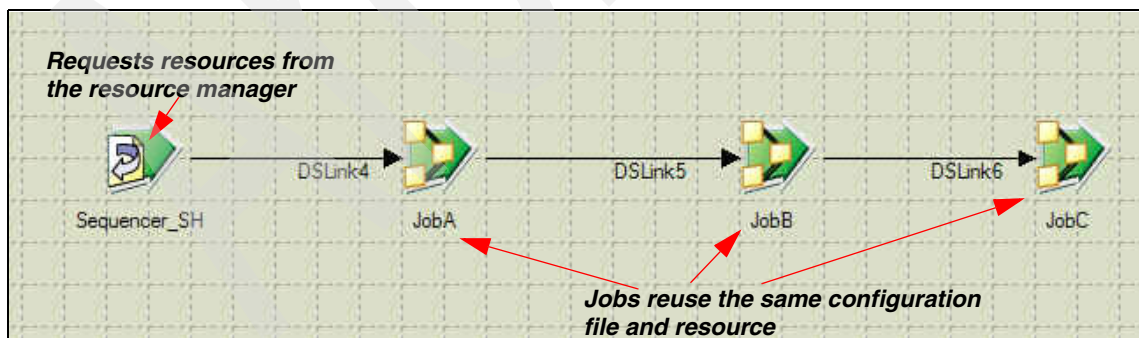


Figure 1-7 Job sequence with sequencer.sh script in the Execute Command Activity stage

Job name: Create_SalesFF

Invocation Id: JobA

Execution action: Run ☐ Do not checkpoint run.

Parameters

Name	Value Expression
\$APT_DUMP_SCORE	
SumKeyStart	
\$APT_GRID_COMPUTENODES	
\$APT_GRID_PARTITIONS	
\$APT_GRID_SEQFILE_HOST	Field(Sequencer_SH.\$CommandOutput," ",1)
\$APT_GRID_SEQFILE_HOST2	Field(Sequencer_SH.\$CommandOutput," ",2)
\$APT_GRID_ENABLE	
SOURCEDIR	
FILE_SUFFIX	
\$APT_CONFIG_FILE	Field(Sequencer_SH.\$CommandOutput," ",3)

Figure 1-8 Passing the three values from the sequencer.sh script output

To create a dynamic configuration file for a job sequence, perform the following steps:

1. Call the sequencer.sh script that is provided with the Grid Enablement Toolkit with the Execute Command Activity stage with the following command line attributes. The sequencer.sh script invokes the Dynamic.sh script.
 - -f <name of new apt_config_file>
 - -c <number of compute nodes>
 - -p <number of partitions per node>

Figure 1-9 on page 21 shows a job sequence with the Execute Command Activity stage added.

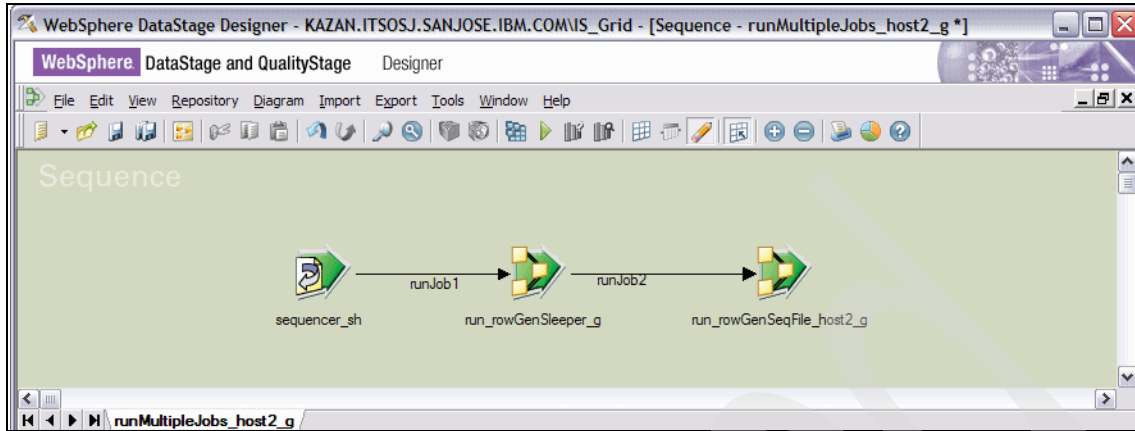


Figure 1-9 Job sequence with a Execute Command Activity stage for the *sequencer.sh* script

Figure 1-10 shows the configuration of this stage with the *sequencer.sh* script and command line attributes in the Parameters field under the **ExecCommand** tab in the Execute Command Activity window.

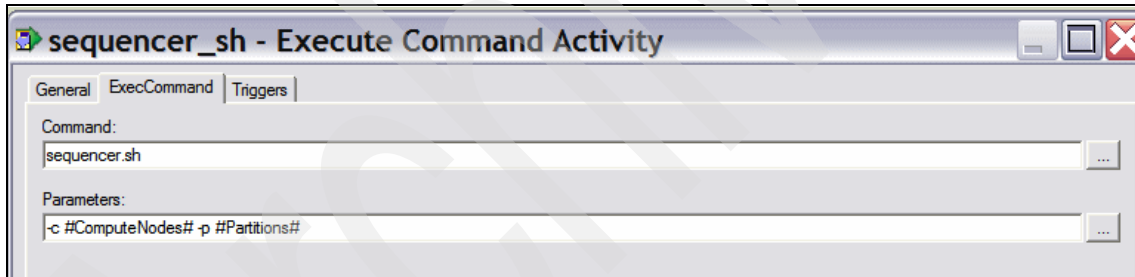


Figure 1-10 Execute Command Activity stage for the *sequencer.sh* script with parameters

2. For each job in the job sequence, set the following parameters:
 - APT_GRID_ENABLE must be set to NO (default).
This setting prevents resources from being allocated to the job twice: once for the sequence, and the second time for the job.
 - APT_CONFIG_FILE must be present.
 - APT_GRID_SEQFILE_HOST and APT_GRID_SEQFILE_HOST2 must be passed if that job has one or more Sequential File stages in it.

Figure 1-11 on page 22 shows the three output values of the *sequencer.sh* script from the Execute Command Activity stage being passed to one of the

jobs in the job sequence. It shows the \$APT_GRID_ENABLE parameter set to NO.

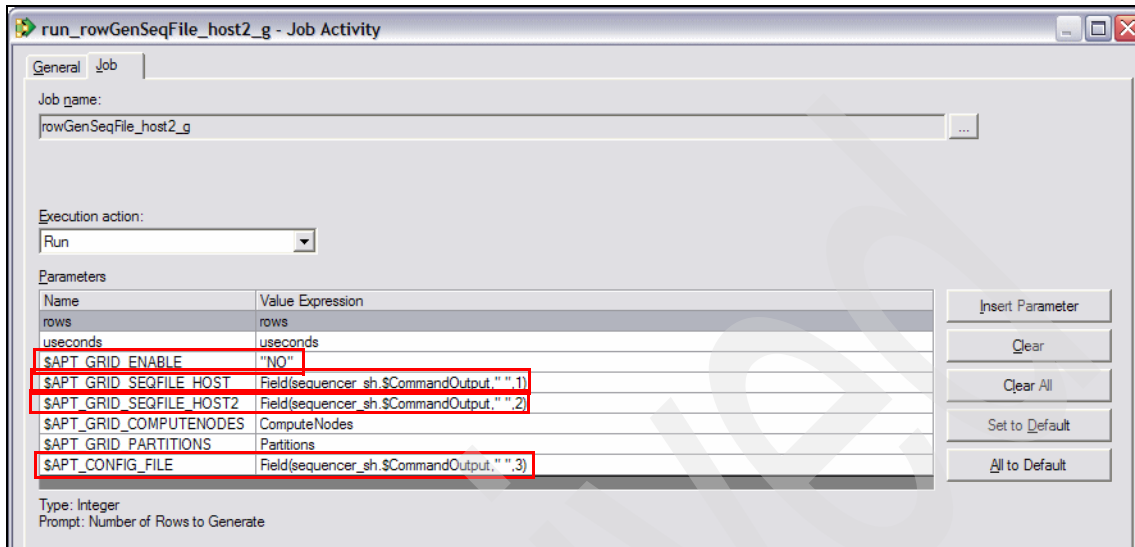


Figure 1-11 Values passed from the sequencer.sh script output to a job in the job sequence

Tip: Requesting resources from the resource manager may sometimes constitute a significant portion of a job's execution time (as much as 30 seconds) especially for short running jobs. For a long series of jobs, the cost of requesting resources can add up to hours if individual jobs are grid-enabled.

If you have several small jobs that run back to back, it is more efficient to call the resource manager once, then run all the jobs in the sequencer with the dynamic configuration file created using the expression: `Field(<stagenam>.$CommandOutput," ",<fieldnum>)` as shown in Figure 1-8 on page 20. Reducing the number of requests to the resource manager reduces idle time, as well as enables you to control resource usage across a series of jobs. The jobs in the sequencer can also run concurrently using the same nodes and resources.

Sequential File processing

The \$APT_GRID_SEQFILE_HOST and \$APT_GRID_SEQFILE_HOST2 environment variables shown in Figure 1-6 on page 18 are only required when Sequential File stages are defined in the job.

If these two parameters are defined on the Parameters tab, then they are set at run time, assuming \$APT_GRID_ENABLE is set to YES at run time.

Note: If running from a job sequencer or other method, these values must be set based on valid values within the \$APT_CONFIG_FILE being used.

Because sequential file I/O cannot be processed on the frontend (conductor) node, all path names must be prefixed by a host name that identifies the host responsible for reading and writing the actual file. Because you do not know the actual host name that is assigned, you must define a variable (\$APT_GRID_SEQFILE_HOST and \$APT_GRID_SEQFILE_HOST2) that will be set by using the Grid Enablement Toolkit:

- ▶ \$APT_GRID_SEQFILE_HOST acts as a place holder for a node (Figure 1-12) allocated by the resource manager at runtime and provides the host name for the Sequential File stage. This is usually the first compute node assigned by the resource manager.
- ▶ \$APT_GRID_SEQFILE_HOST2 is usually the second compute node assigned by the resource manager.

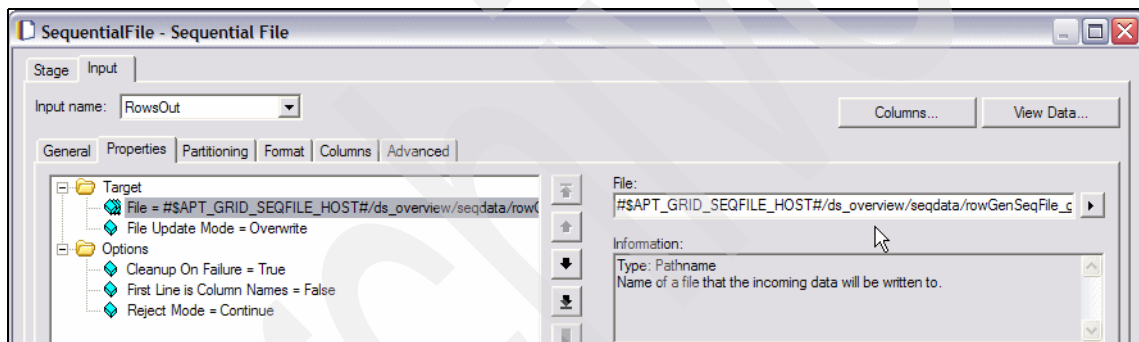


Figure 1-12 \$APT_GRID_SEQFILE_HOST variable in the Sequential File Properties tab

When run on a single job, the Grid Enablement Toolkit modifies the contents of these parameters based on the first two nodes assigned by the resource manager. Each parameter includes an actual host name followed by a colon such as "phoenixpvt.itsosj.sanjose.ibm.com:." If there's only one node assigned by the resource manager, then both the variables are assigned the same host name value.

If you run job sequences to run multiple jobs using the same generated configuration file, you must pass the \$APT_GRID_SEQFILE_HOST and \$APT_GRID_SEQFILE_HOST2 values assigned during the initial creation of the configuration file into each stage.

Information Analyzer enablement

To grid-enable Information Analyzer jobs, you must modify the project defaults with the following information

- ▶ `$APT_GRID_ENABLE: YES`
- ▶ `$APT_GRID_PARTITIONS: 6`

This value maximizes parallelism on each compute node

1.4.3 Execution flow

Figure 1-13 on page 25 shows the invocation of the different scripts and executables when an InfoSphere Information Server application is submitted for execution.

1. The intercept script `osh` gets invoked, which calls the `osh_conductor.sh` script.
2. The `osh_conductor.sh` script performs some initial validation of rules and determines if a dynamic configuration as per the following rules:
 - If `$APT_GRID_ENABLE` is set to `NO`, then the renamed `osh.exe` is executed using the specified configuration file. This corresponds to using an existing configuration file created through another method.
 - If `$APT_GRID_ENABLE` is set to `YES`, then the `Dynamic_Grid.sh` script is invoked.
3. The `Dynamic_Grid.sh` script interacts with the resource manager and generates a dynamic configuration file when resources required by the job become available.

Note: In the case of a job sequence, the `Dynamic_Grid.sh` script may also be invoked directly by `sequencer.sh` as shown in Figure 1-13 on page 25.

4. The `Dynamic_Grid.sh` script then passes the dynamically generated configuration file to the `osh.exe` executable. The `Dynamic_Grid.sh` script monitors the execution of the job or job sequence. When the job completes, the `Dynamic_Grid.sh` script performs a cleanup of the job directory and terminates.

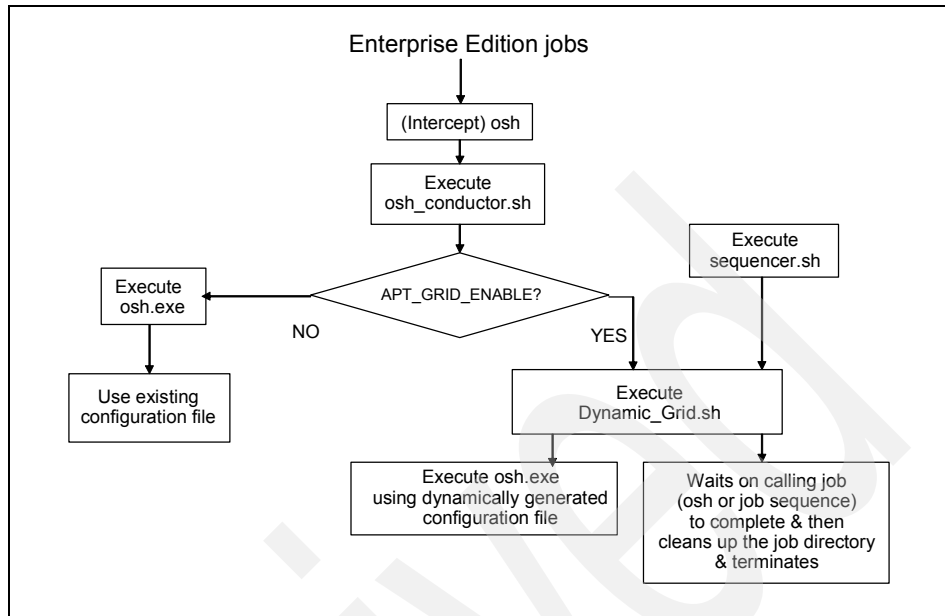


Figure 1-13 Invocation of the *Dynamic_Grid.sh* script

Figure 1-14 on page 26 describes in more detail the following steps performed by the *Dynamic_Grid.sh* script:

1. Calls the *GenSubmit.Class* that creates the following scripts:
 - *<jobname>_PID.wait.sh*
 - *<jobname>_PID.sh*
2. Initiates the *<jobname>_PID.wait.sh* script process in the background (nohup).
3. Submits *<jobname>_PID.sh* to the resource manager specified in the *\$GRIDHOME/grid_global_values Engine=<resource_manager>* setting.
4. Waits on the *<jobname>_PID.sh* to create the configuration file by monitoring for the *<jobname>_PID_start* file in the shared work directory.
 - a. *<jobname>_PID.sh* (that the resource manager schedules on a compute node) performs the following:
 - i. Waits for resources to become available and then creates the configuration file in the shared work directory based on nodes assigned by the resource manager, and holds the assigned resources.
 - ii. Creates a file called *<jobname>_PID_start* in the shared work directory that is a signal to *Dynamic_Grid.sh* that it can resume processing.

- iii. Waits for the <jobname>_PID_end signal file in the shared work directory. When it sees that file, it releases the held resources and terminates.
5. When Dynamic_Grid.sh sees the <jobname>_PID_start signal file in the shared work directory, it retrieves the dynamically generated configuration file from the shared work directory and submits osh.exe with this file.
6. Waits for the process that called Dynamic_Grid.sh (job or sequencer.sh) to complete and then terminates.

The waiting <jobname>_PID.wait.sh process monitors the process id of the Dynamic_Grid.sh script instance and when it determines that the process is no longer alive, it creates a <jobname>_PID_end signal file and terminates. As mentioned earlier, the <jobname>_PID.sh releases all the held resources after seeing <jobname>_PID_end file in the shared work directory and terminates.

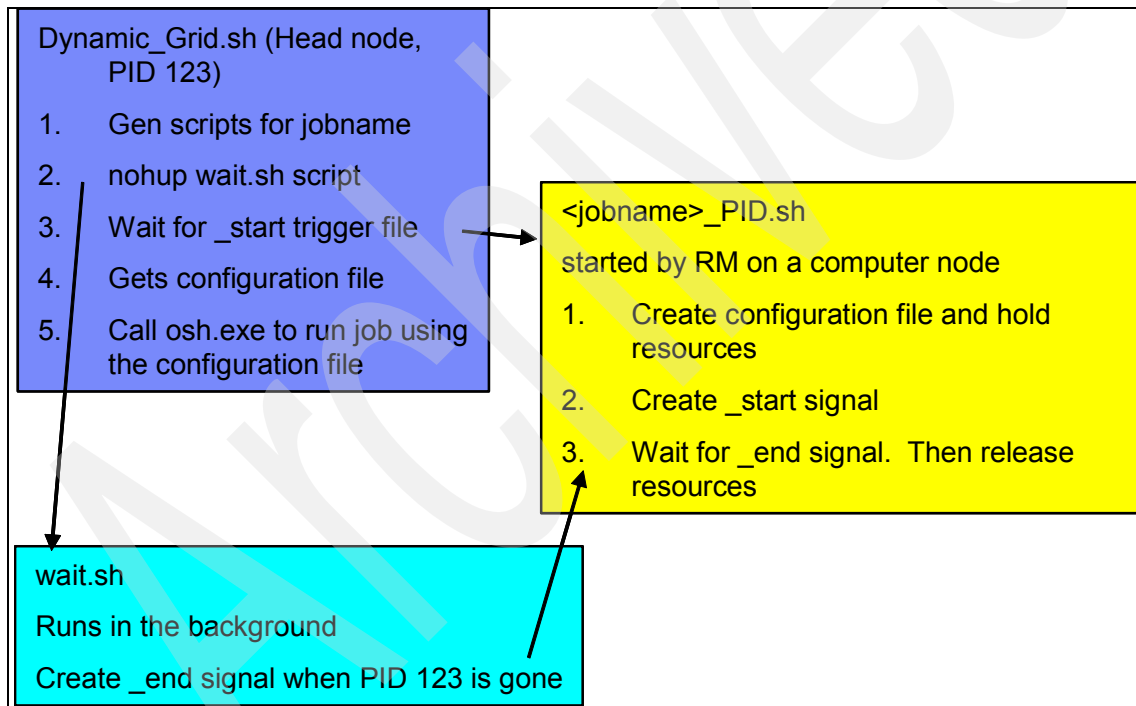


Figure 1-14 Dynamic_Grid.sh execution flow

1.4.4 Grid computing versus cluster computing

The Parallel Framework supports both clustered and grid deployment of DataStage and IBM InfoSphere QualityStage jobs by allowing a single parallel job to run across multiple servers.

In both the clustered deployment and grid deployment scenarios, one server is designated as the primary node. This is the node against which InfoSphere Information Server clients validate engine credentials. It is also where the other engine tier components (such as Job Monitor) execute.

Cluster computing

In a clustered scenario, the APT configuration files for the Parallel Framework are statically defined and referenced explicitly in the job. If a single node in the configuration file fails, you must modify the APT configuration file and replace the name of the failing node with another one for the job to proceed. Node use generally becomes an issue because the explicit specification of a configuration file drives node use. If the same configuration file is used multiple times in different jobs, the specified nodes get overused while those not named in the configuration file might be idle. Management of node use can be a significant effort in clustered configurations.

The submission of job `grid_J09_IL_LoadLookupCustomerDim` in a clustered configuration follows this process:

1. When the `grid_J09_IL_LoadLookupCustomerDim` job (Figure 1-15 on page 28) is submitted, the Job Run Options window opens.
2. `mpp_3nodes_2PN.apr` (Figure 1-16 on page 28) is the name of the configuration file to be used and for the grid to be disabled (ENABLE.... is NO).
3. Figure 1-17 on page 29 shows the successful execution of the `grid_J09_IL_LoadLookupCustomerDim` job.
4. In the Director log contents of this job (Figure 1-18 on page 29), you see the named configuration file (`mpp_3nodes_2PN.apr` highlighted in the figure) that is used in the execution of this job.
5. The Event Detail output log (Figure 1-19 on page 30) shows the parameters that are used by this job.

6. The contents of the mpp_3nodes_2PN.apr configuration file (Figure 1-20 on page 30 and Figure 1-21 on page 31) explicitly identifies the six nodes to be used as follows:
- node1 and node2 corresponding to the kazanpvt.itsosj.sanjosel.ibm.com server
 - node3 and node4 corresponding to the taruspvt.itsosj.sanjosel.ibm.com server
 - node5 and node6 corresponding to the phoenixpvt.itsosj.sanjosel.ibm.com server

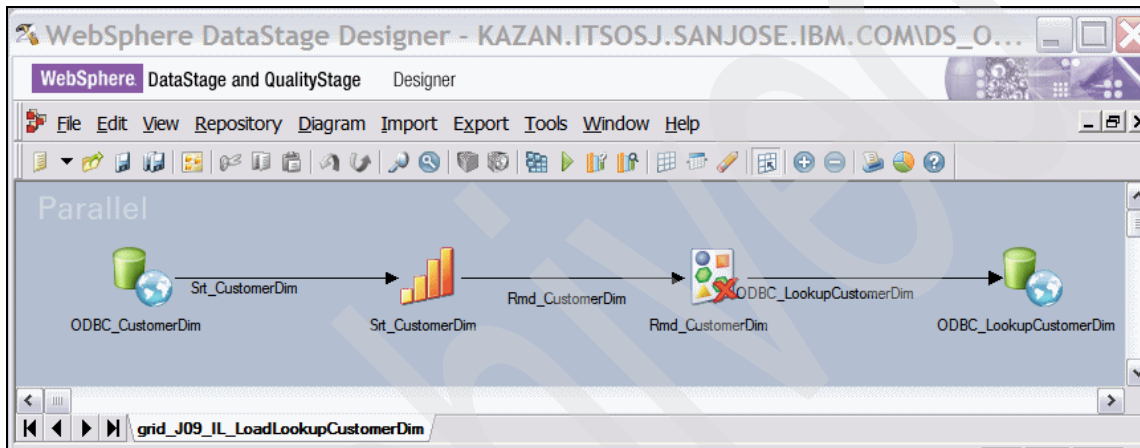


Figure 1-15 Cluster configuration, part 1 of 7

The screenshot shows the 'grid_J09_IL_LoadLookupCustomerDim - Job Run Options' dialog box. The 'Parameters' tab is selected. The dialog contains a table with the following data:

Name	Value
Configuration file	/opt/IBM/InformationServer/Server/Configurations/mpp_3nodes_2PN.apr
ENABLE - Use DYNAMIC_CONFIG_SCRIPT (NO/YES/OLD)	NO
COMPUTENODES - Number of Compute Nodes	1
PARTITIONS - Number of partitions per Node (compute)	1
Report score	False

Buttons on the right side of the dialog include 'Set to Default', 'All to Default', and 'Property Help'. At the bottom of the dialog are buttons for 'Run', 'Validate', 'Cancel', and 'Help'.

Figure 1-16 Cluster configuration, part 2 of 7

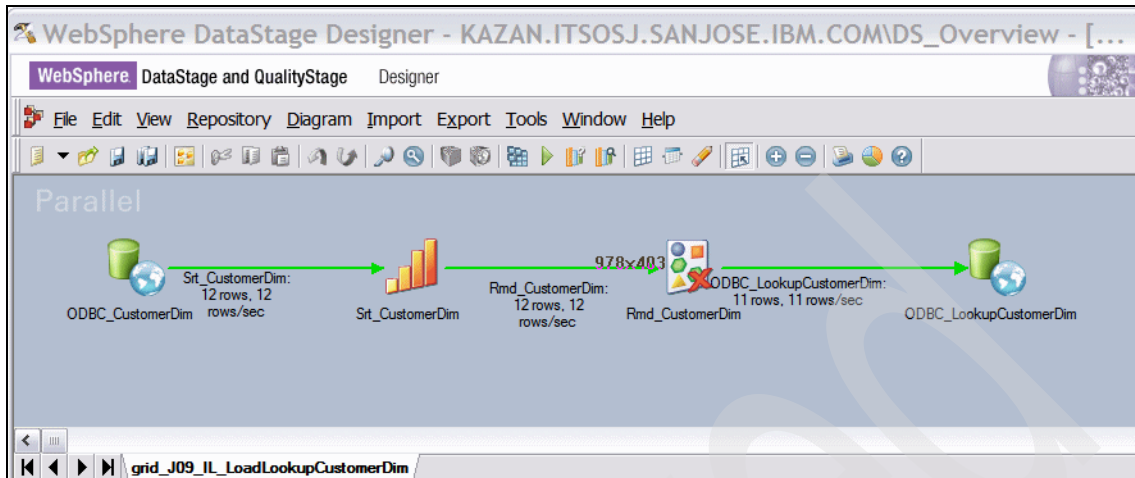


Figure 1-17 Cluster configuration, part 3 of 7

WebSphere DataStage Director - KAZAN.ITSOSJ.SANJOSE.IBM.COM\DS_Overview

WebSphere DataStage and QualityStage Director

Project View Search Job Tools Help

>Occurred	>On date	Type	Event
10:14:25 AM	4/24/2008	Control	Starting Job grid_J09_IL_LoadLookupCustomerDim. (...)
10:14:26 AM	4/24/2008	Info	Environment variable settings: (...)
10:14:26 AM	4/24/2008	Info	Parallel job initiated
10:14:26 AM	4/24/2008	Info	OSH script (...)
10:14:27 AM	4/24/2008	Info	<osh_conductor> Checking Authorization...
10:14:27 AM	4/24/2008	Info	<osh_conductor> Authorized to proceed.
10:14:27 AM	4/24/2008	Info	main_program: IBM WebSphere DataStage Enterprise Edition 8.0.1.4668 (...)
10:14:27 AM	4/24/2008	Info	main_program: orchgeneral: loaded (...)
10:14:28 AM	4/24/2008	Info	ODBC_LookupCustomerDim: [IIS-CONN-ODBC-000105] Connected to UDB DB2 for Windows, UNIX, and Linux, version 09.13.1
10:14:28 AM	4/24/2008	Info	main_program: APT configuration file: /opt/IBM/InformationServer/Server/Configurations/mpp_3nodes_2PN.apl (...)
10:14:29 AM	4/24/2008	Warning	ODBC_LookupCustomerDim: When checking operator: When binding input interface field 'CUSTOMER_ID' to field 'CUSTOMER_ID'...
10:14:29 AM	4/24/2008	Info	ODBC_CustomerDim.0: [IIS-CONN-ODBC-000105] Connected to UDB DB2 for Windows, UNIX, and Linux, version 09.13.0000t
10:14:29 AM	4/24/2008	Info	ODBC_LookupCustomerDim.4: [IIS-CONN-ODBC-000105] Connected to UDB DB2 for Windows, UNIX, and Linux, version 09.13.0000t
10:14:29 AM	4/24/2008	Info	ODBC_LookupCustomerDim.5: [IIS-CONN-ODBC-000105] Connected to UDB DB2 for Windows, UNIX, and Linux, version 09.13.0000t
10:14:29 AM	4/24/2008	Info	ODBC_LookupCustomerDim.2: [IIS-CONN-ODBC-000105] Connected to UDB DB2 for Windows, UNIX, and Linux, version 09.13.0000t
10:14:29 AM	4/24/2008	Info	ODBC_LookupCustomerDim.3: [IIS-CONN-ODBC-000105] Connected to UDB DB2 for Windows, UNIX, and Linux, version 09.13.0000t
10:14:29 AM	4/24/2008	Info	ODBC_LookupCustomerDim.1: [IIS-CONN-ODBC-000105] Connected to UDB DB2 for Windows, UNIX, and Linux, version 09.13.0000t
10:14:29 AM	4/24/2008	Info	ODBC_LookupCustomerDim.0: [IIS-CONN-ODBC-000105] Connected to UDB DB2 for Windows, UNIX, and Linux, version 09.13.0000t
10:14:29 AM	4/24/2008	Info	ODBC_LookupCustomerDim.3: [IIS-CONN-DAAPI-000336] Number of rows updated: 0 (...)
10:14:29 AM	4/24/2008	Info	ODBC_LookupCustomerDim.2: [IIS-CONN-DAAPI-000336] Number of rows updated: 1 (...)
10:14:29 AM	4/24/2008	Info	ODBC_LookupCustomerDim.4: [IIS-CONN-DAAPI-000336] Number of rows updated: 1 (...)
10:14:29 AM	4/24/2008	Info	ODBC_LookupCustomerDim.5: [IIS-CONN-DAAPI-000336] Number of rows updated: 3 (...)
10:14:29 AM	4/24/2008	Info	ODBC_LookupCustomerDim.1: [IIS-CONN-DAAPI-000336] Number of rows updated: 3 (...)
10:14:29 AM	4/24/2008	Info	ODBC_LookupCustomerDim.0: [IIS-CONN-DAAPI-000336] Number of rows updated: 3 (...)
10:14:29 AM	4/24/2008	Info	main_program: Step execution finished with status = OK.
10:14:29 AM	4/24/2008	Info	main_program: Startup time, 0:02; production run time, 0:00.
10:14:29 AM	4/24/2008	Info	Parallel job reports successful completion
10:14:29 AM	4/24/2008	Control	Finished Job grid_J09_IL_LoadLookupCustomerDim.

Figure 1-18 Cluster configuration, part 4 of 7

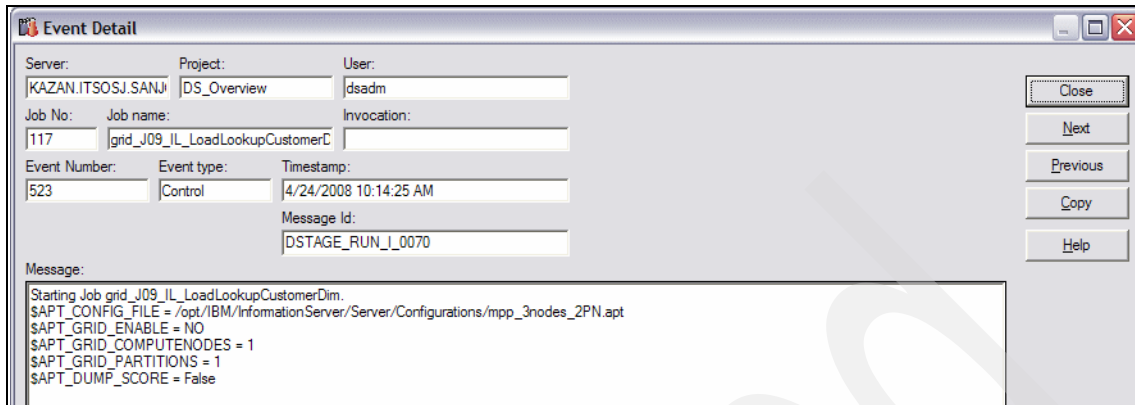


Figure 1-19 Cluster configuration, part 5 of 7

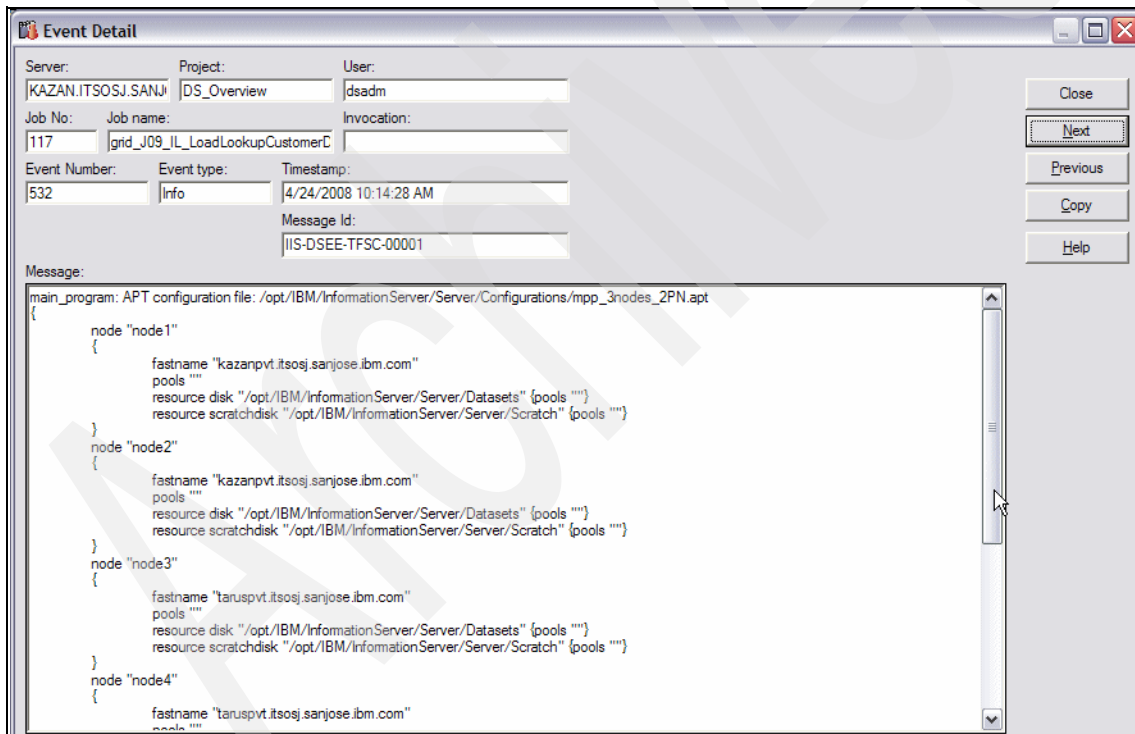


Figure 1-20 Cluster configuration, part 6 of 7

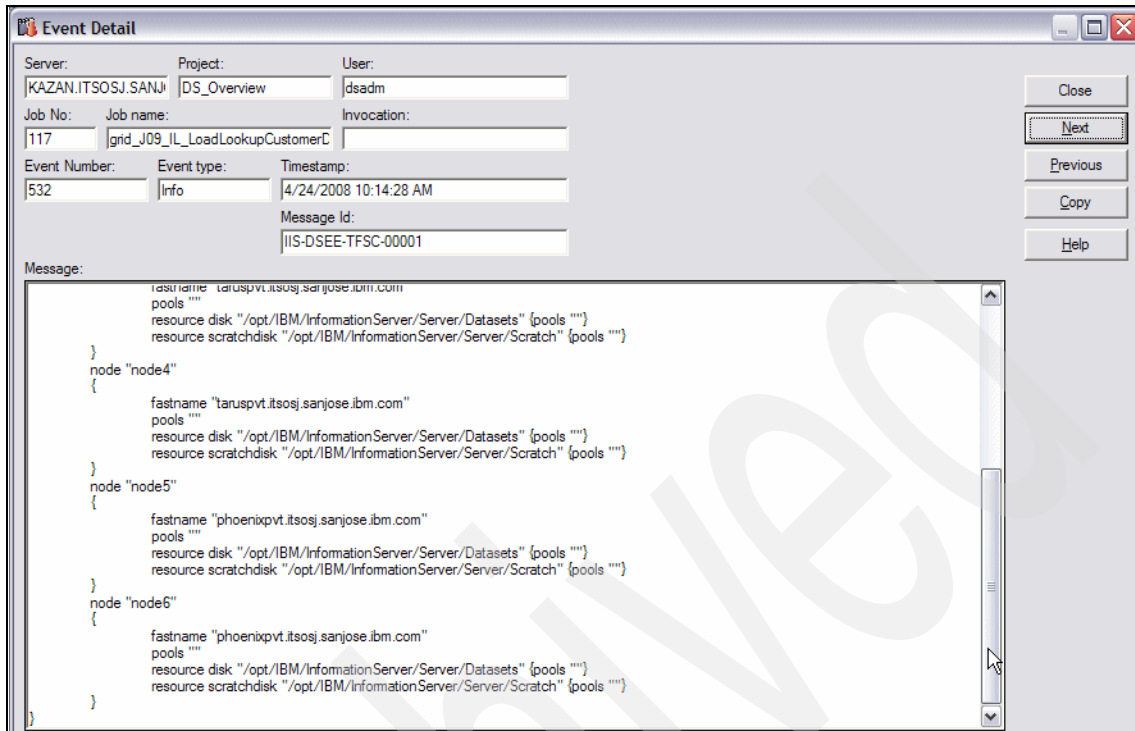


Figure 1-21 Cluster configuration, part 7 of 7

Grid computing

With a grid implementation, the static configuration file normally defined for a cluster is replaced with a dynamic configuration file created at runtime. This is accomplished with the Grid Enablement Toolkit (described in “Grid Enablement Toolkit” on page 247) provided. The dynamic configuration files are created based on using a resource manager that identifies idle servers as described in 1.3.1, “Main components” on page 7.

The resource manager queues submitted jobs until the required resources become available. When the node or nodes requested are available, a configuration file is dynamically generated for the submitted job. While the job is running, the resource manager prevents additional jobs from using the same resources until the running job completes. The Grid Enablement Toolkit is used to provide an interface into the resource manager that allows the current run methods for InfoSphere Information Server applications to be used with minimal impact.

Figure 1-22 on page 33 through Figure 1-27 on page 36 show the same job `grid_J09_IL_LoadLookupCustomerDim` submitted this time with grid enabling as follows:

1. After the job `grid_J09_IL_LoadLookupCustomerDim` is submitted (Figure 1-15 on page 28), the Job Run Options window opens.
2. Figure 1-22 on page 33 shows the `mpp_3nodes_2PN.aprt` configuration file being used with the grid enabled (`ENABLE....` is YES), and the `COMPUTENODES = 3` and `PARTITIONS = 2`.
3. Figure 1-23 on page 33 shows the successful execution of this job.
4. The highlighted portion of the Director log contents of this job (Figure 1-23 on page 33) shows the generated configuration file (`grid_J09_IL_LoadLookupCustomerDim_27680.config.aprt`) being used in the execution of this job even though `mpp_3nodes_2PN.aprt` was specified in the Job Run Options. It also shows the three compute nodes `luxorpvt.itsosj.sanjose.ibm.com`, `phoenixpvt.itsosj.sanjose.ibm.com`, and `taruspvt.itsosj.sanjose.ibm.com`.

This uses different nodes. The conductor node `kazanpvt.itsosj.sanjose.ibm.com` is not involved.

5. The Event Detail output log (Figure 1-24 on page 34) shows the parameters used by this job.
6. The `grid_J09_IL_LoadLookupCustomerDim_27680.config.aprt` configuration file (Figure 1-25 on page 34 and Figure 1-26 on page 35) explicitly identifies the six nodes used as follows:
 - Compute1 and Compute2 correspond to the `luxorpvt.itsosj.sanjose.ibm.com` server
 - Compute3 and Compute4 correspond to the `phoenixpvt.itsosj.sanjose.ibm.com` server
 - Compute5 and Compute6 corresponding to the `taruspvt.itsosj.sanjose.ibm.com` server

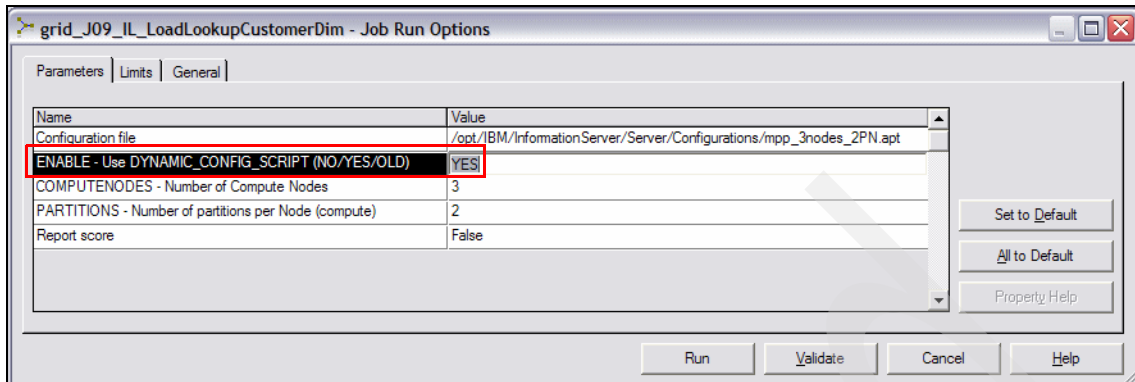


Figure 1-22 Grid configuration, part 1 of 6

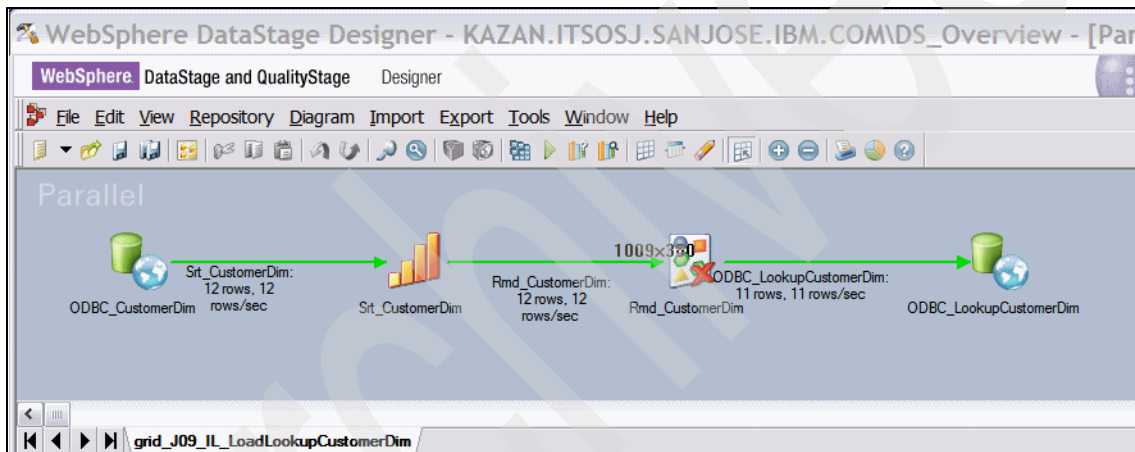


Figure 1-23 Grid configuration, part 2 of 6

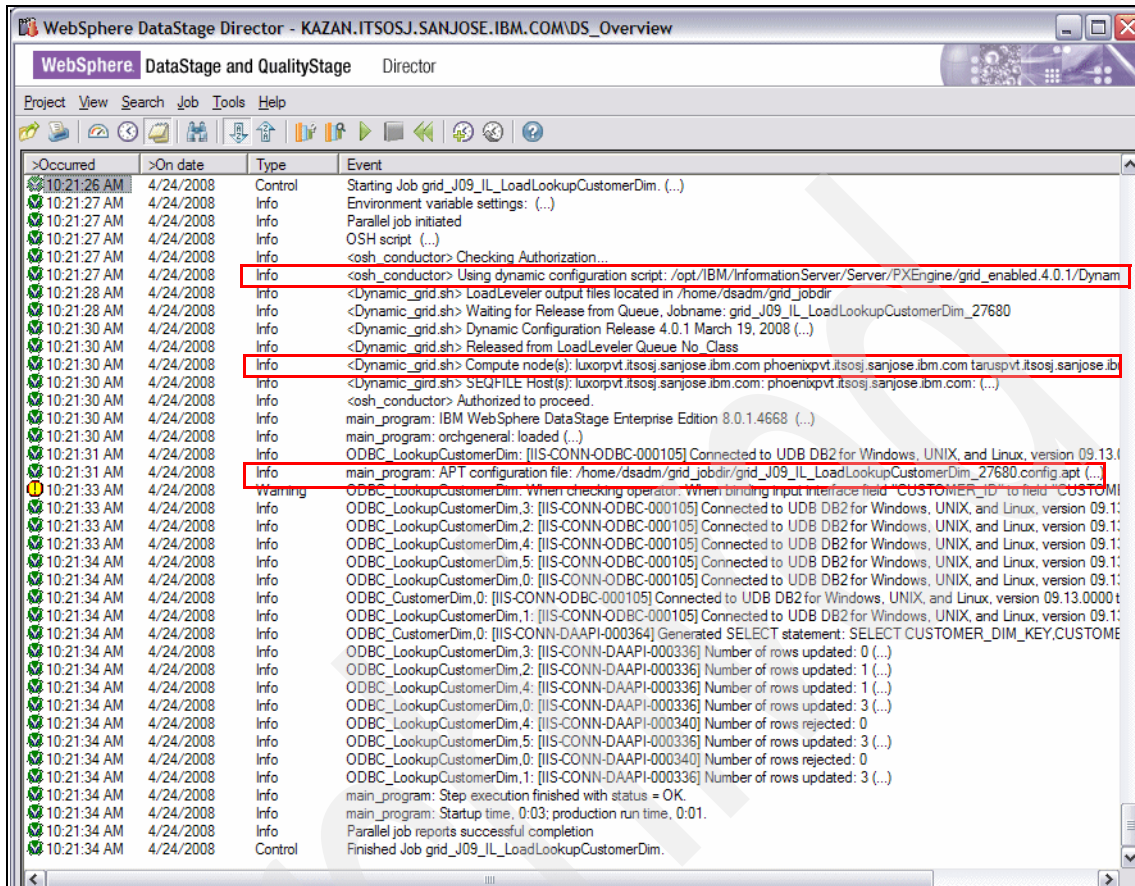


Figure 1-24 Grid configuration, part 3 of 6

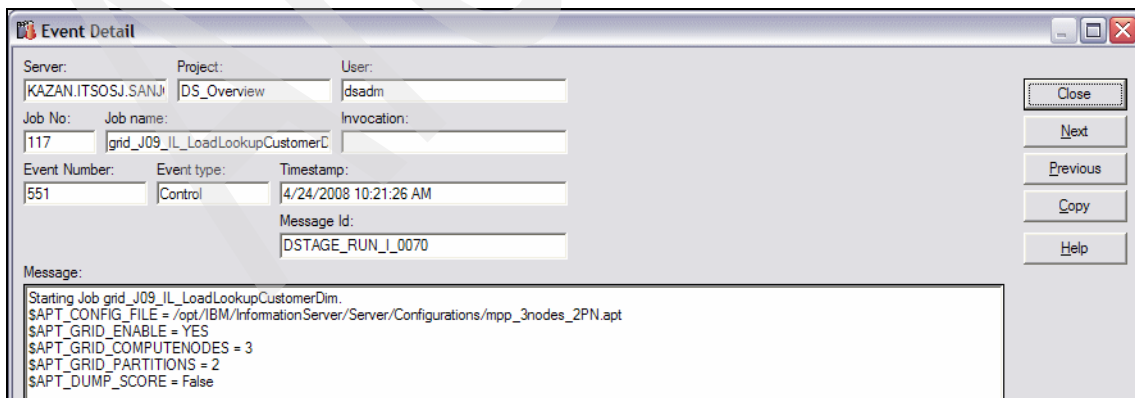


Figure 1-25 Grid configuration, part 4 of 6

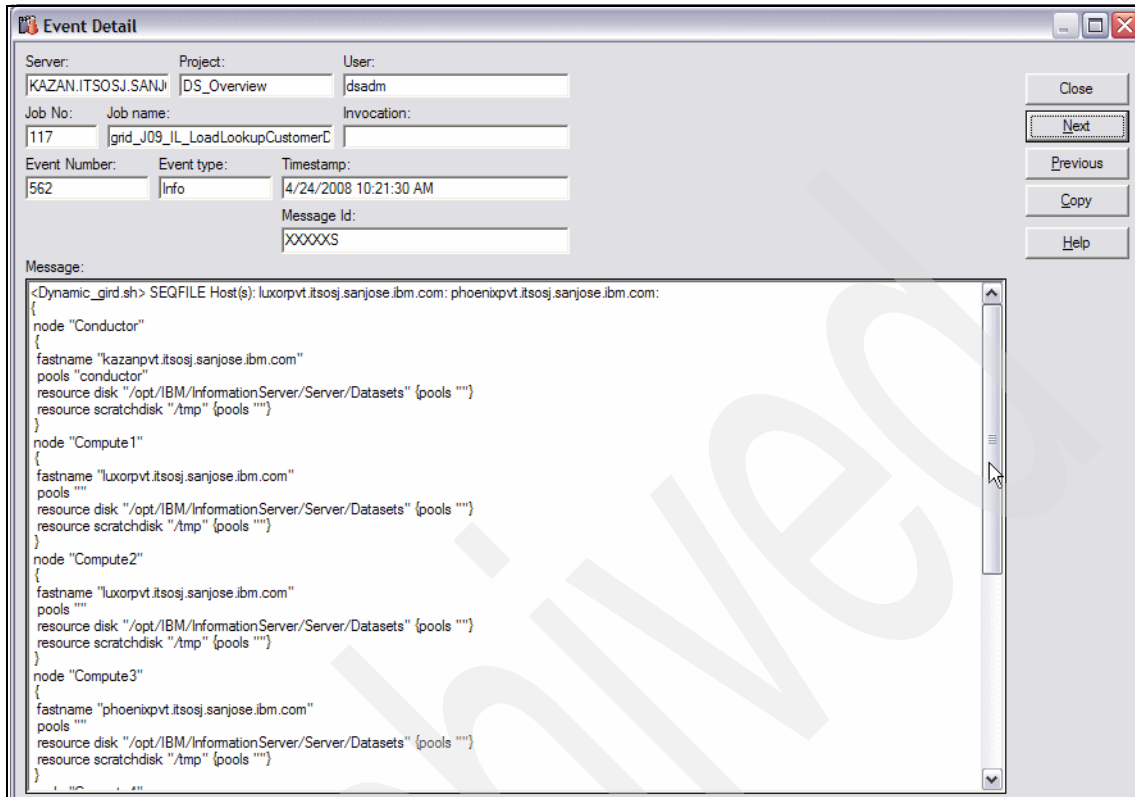


Figure 1-26 Grid configuration, part 5 of 6

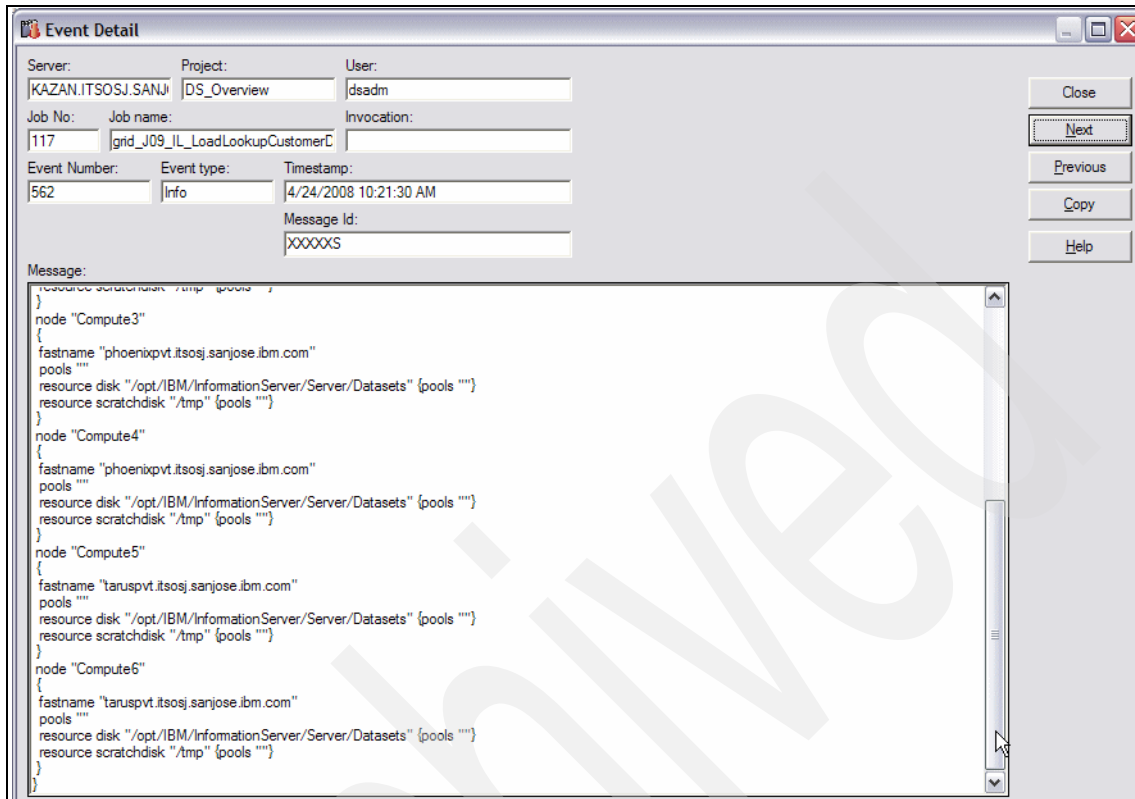


Figure 1-27 Grid configuration, part 6 of 6

1.5 Managing the InfoSphere Information Server grid environment

The grid environment is a complex one that must be managed effectively to ensure that users' service levels are met. This includes managing the resource manager, monitoring and tuning the grid environment for performance, and troubleshooting problems.

1.5.1 Resource manager

The resource manager enables the InfoSphere Information Server grid computing solution to use a resource (node) without knowing which resource (node) is providing the service. This allows a process to use a resource (node) today that was not available yesterday, or that may become unavailable tomorrow.

The resource manager supports this scenario by keeping track of resources, identifying the servers that are down, and monitoring the system load.

The Grid Enablement components provide for multiple resource managers to be selected including IBM LoadLeveler, Altair PBSpro, Sun™ Grid Engine (SGE), Torque (enhanced version of open PBS), Platform LSF, Condor and DataSynapse GridServer.

The supported resource managers provide options that we do not necessarily use, but each does provide the components to run the parallel framework of InfoSphere Information Server on multiple nodes.

The resource manager can be used to limit the jobs by project through queues. The queues provide the following management capabilities:

- ▶ Act as a license restrictor
You can restrict the maximum number of jobs that can be started for any queue. For example, you might only license 8 CPUs for Information Analyzer, but the grid has 20 nodes (machines), each with two CPUs. Thus, you can use any four nodes out of the 20 available, and the resource manager limits the number of concurrent Information Analyzer jobs.
- ▶ Limit jobs for given projects based on the time of day
- ▶ Can provide priority over other queues

The options of a particular resource manager determine the specific management capabilities. Refer to the relevant product documentation for details.

1.5.2 Monitoring and tuning

The critical items to monitor are the CPU, network, and memory use as it relates to the following factors:

- ▶ The performance of a job
- ▶ How a job uses the nodes it is running on
- ▶ How all the nodes in the grid environment are being used

Monitoring a single machine can be performed using commands such as **top**, **sar**, and **vmstat**. To view what is occurring across all the machines requires connecting to the individual machine. A number of third party monitoring tools are available such as Ganglia that can provide a holistic view of the use across all the nodes.

When a performance problem is found, it must be tuned.

A discussion of this topic is beyond the scope of this Redbooks publication.

IBM provides a performance tuning workshop, details of which can be obtained at the following Web page:

http://www-306.ibm.com/software/sw-library/en_US/detail/0184470D93565M95.html

1.5.3 Troubleshooting

In this section, we identify the common problems encountered in customer environments. Most of the problems identified here are usually not encountered if the steps identified in the Grid Enablement Toolkit documentation (see Appendix A, “Grid deployment toolkits” on page 243) are followed.

- ▶ Jobs do not get released for execution from the resource manager
This is most likely because the resource manager's environment variables (including the path for the executables) are not defined for the process submitting the actual job. To resolve the problem, perform the following steps:
 - a. Make sure that the resource manager path and settings are defined for the process
 - b. Restart the InfoSphere Information Server engine processes on the front end node using the **uv -admin -stop/start** commands
 - c. Use the command **l1stq** to show jobs within queues. Details on the job can be obtained by issuing the command **l1stq <jobid>**. This identifies why the job is not being started. The **l1stq** command comes with Grid Enablement Toolkit and works with IBM Tivoli Workload Scheduler LoadLeveler and SGE.

- ▶ Resource Manager indicates that it is running, but no dynamic APT configuration file has been created.

This is a common problem when users in the grid are not configured correctly. The correct way to set up users is by using NIS (or something similar) with the user's home directory as an NFS mount to the compute nodes.

In addition, the environment variable settings for each user (.profile) must include the necessary components to dynamically generate the `APT_CONFIG_FILE`, the most important component being java.

If the configuration does not use as single mount point for the user, then it is more than likely to cause this problem. Make sure when you test the connection to the compute nodes (through ssh), that the `PATH` to the Java library is correct and that `<JOBDIR>` for the jobs exists on the actual compute node.

How to see what jobs are queued and why they may be waiting: Each resource manager has its own way of reviewing the queues to determine what is running or waiting. In every case, the actual job runs on the compute node and not the front end node.

There is a process started on the front end node used to interact with the resource manager and the application. Use the `listq` command (currently available for LoadLeveler and SGE) to see the jobs defined within the resource manager queue.

1.6 Successful practices for the InfoSphere Information Server grid environment

In this section, we provide a brief overview of successful practices gained from our experiences with deploying InfoSphere Information Server in a grid environment on customer engagements. A detailed discussion of the considerations involved is beyond the scope of this Redbooks publication.

The following section describes these successful practices:

- ▶ “Configuration guidelines” on page 40
- ▶ “Tuning guidelines” on page 42

1.6.1 Configuration guidelines

The following guidelines are based on field experiences. In some cases, the recommendations documented here are different than those in the installation documentation (that defines minimum values).

The following sections describe the recommendations for the network, front end node (CPU, memory, and disk storage), and compute nodes (CPU, memory, and disk storage).

Network

The Parallel Framework is designed to run on multiple nodes within a grid configuration. To accomplish this, data is distributed between nodes based on business rules (aggregation, joins, and removing duplicates) that can cause network bottlenecks if many concurrent jobs are running.

To prevent such bottlenecks, the grid must be configured with, at minimum, a single 1 GB Private Ethernet switch. Having multiple 1 GB network switches for public (NAS data and database access) and private (messages and data between nodes) networks provides the best solution.

Front end node

InfoSphere Information Server components run on different hardware configurations, including i386 or i586 platforms and the IBM System x™ servers. The operating system being loaded must be 32-bit compatible. A compatibility of 64 bit is allowed as long as it supports 32-bit applications.

The recommendations for the front end node are as follows:

- CPUs

Minimum two CPUs: More CPUs are required if the front end node is acting as a NAS server to the compute nodes. With InfoSphere Information Server Version 8.x, four CPUs are recommended because of WebSphere® Application Server and DB2 for Linux, UNIX, and Windows.

► **Memory**

A minimum of 8 GB of memory is required on the front end node.

Add additional memory for the front end node if the following circumstances apply to your grid environment:

- If the front end node is acting as a NAS server to the compute nodes (as shown in Figure 1-3 on page 12), add additional memory for NFS to buffer file system activity.
- If you run grid-enabled jobs with stages that have to be front end node constrained, for non-grid-enabled jobs that run on the front end node, such as DataStage Server jobs, which will not run in the grid.

► **Disk storage**

The following recommendations are for disk storage:

- 10 GB for Information Server (WAS / DB / Engine / Logs).
- 25 MB free space in the /var directory.
- Sufficient storage space for any data that is to be held in persistent files (Datasets, Filesets, and sequential files) when the front end node is used as an NFS server to compute nodes.
- Additional space to allow for temporary data storage while a DataStage job is running (scratch space).

Compute nodes

The recommendations are as follows:

- **Minimum of three compute nodes in the configuration:** Less than that must be a cluster.

► **CPUs**

Two CPUs per node.

► **Memory**

The recommended memory for each node is 4 GB. This accommodates the potential for large jobs or lookups that normally exceed 2 GB per node.

Although 4 GB is the recommended memory configuration for compute nodes, it is possible that 2 GB will work in some cases without a performance impact.

Adding additional memory in excess of 4 GB is not cost effective for compute nodes. If the job running requires additional memory, consider using additional compute nodes (partitions per node must decrease).

► Disk storage

The recommendations are as follows:

– Scratch space

Minimum 20 GB

May require additional storage based on sorts and other scratch file usage. It must be a local disk, not NAS-mounted or NFS-mounted. Using a local disk reduces the network requirements for scratch processing.

– NFS mounts from the front end, including Software, Users, and Project directories

Note: If there are NFS mounts of data from the front end node in a SAN-based configuration where the NAS Server is collocated on the front end node as shown in Figure 1-3 on page 12, make sure that the front end node has several NIC cards to distribute data between network cards. If this is not the case, you will be limited to the 1 GB I/O bandwidth from the front end node to all compute nodes.

1.6.2 Tuning guidelines

We recommend the following procedure to determine the number of compute nodes and partitions per compute node for each grid-enabled job in your grid environment:

1. Start with one compute node with one partition for a particular job.

If you have some other basis (such as an existing cluster implementation) to start with a higher number of compute nodes and partitions, then start with that number.

2. Establish a base line performance for this setting.

3. Increase the number of partitions until the CPUs on the node are fully used.

4. Establish whether the runtime performance is satisfactory. It must show continual performance improvement over the base line performance.

– If the runtime performance is not satisfactory, then add another compute node for the particular job.

– The number of partitions on nodes usually remains constant.

Your resource monitor of choice must be used to measure the CPU use of the compute nodes assigned by the resource manager to a particular job. The DataStage and QualityStage Director log identifies the compute nodes assigned to a particular job.

1.7 Setting up the InfoSphere Information Server grid-computing environment

Figure 1-28 provides an overview of the steps in setting up your Information Server grid computing environment. Each of these steps is briefly described after the figure.

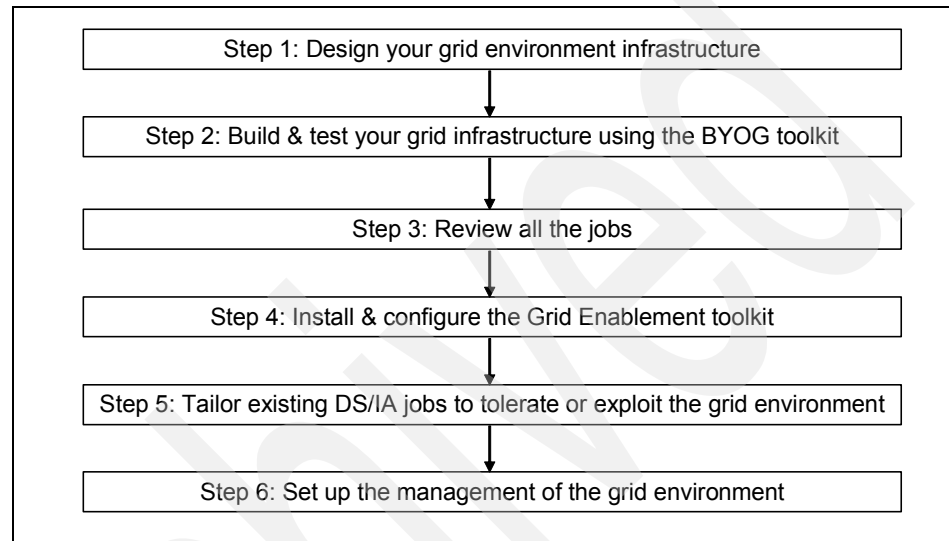


Figure 1-28 Steps to set up your InfoSphere Information Server grid environment

1.7.1 Step 1: Designing your grid environment infrastructure

After choosing to implement a grid environment as a solution to your business requirements, you must decide on the following elements of the grid infrastructure:

- Operating system (OS) for the compute nodes

If you have an existing InfoSphere Information Server, then that OS platform influences the decision, because all OSs must be the same (homogeneous) in the grid environment.

Note: The Build Your Own Grid (BYOG) toolkit described in “Build Your Own Grid toolkit” on page 244 is limited to RedHat Enterprise Linux and SUSE Linux only. However, the Grid Enablement Toolkit described in “Grid Enablement Toolkit” on page 247 works on any platform, but its primary focus is Linux.

- ▶ Number of compute nodes in the grid environment

A starting point of three or more nodes should be based on the following considerations:

- Number of concurrent jobs or job sequences

Each concurrent job and job sequence requires its own set of compute nodes. The cumulative of all these compute nodes is what is required as a starting point. For example, concurrent Job A requires three compute nodes, concurrent Job B requires four compute nodes, and concurrent JobSequence C requires three compute nodes, a total of 10 (3 + 4 + 3) compute nodes are required for the total of three concurrent tasks to run.

By implementing job sequences that combine multiple jobs, you can design your own concurrent/parallelism within a job sequence and thereby reduce the number of compute nodes required. A discussion of designing job sequences with user-defined parallelism is beyond the scope of this Redbooks publication. IBM offers a services engagement to help you with this process, however, and you must contact your local IBM sales representative for details on engaging these services.

- Number of projects running concurrent jobs
- Total data volume and time constraints required to source, transform, and load the target data.

A higher number may be appropriate if you currently have a cluster implementation with more than three servers.

- ▶ Front end node capacity as described in “Front end node” on page 40
- ▶ Compute node capacity as described in “Compute nodes” on page 41
- ▶ Whether to use a HA solution

If an HA solution is desired, the question becomes whether a dedicated or shared standby node configuration is appropriate.

If a shared standby node is selected, the question becomes whether this must be one of the compute nodes. This requires the compute node to have the identical capacity of that of the front end node.

- ▶ Whether to use a NAS-based configuration or a SAN-based configuration

Your existing IT infrastructure will most likely influence this decision.

- ▶ Whether to have a separate (private) network for the connections between the front end node and the compute nodes

This decision will determine whether additional NICs are required.

- ▶ The resource manager to use in the grid environment
Cost and functionality (such as multiple queue support) will determine this decision.
- ▶ The resource monitor to use in the grid environment
Cost and functionality will determine this decision.
- ▶ Whether to use the PXE boot process for configuring the compute nodes

Compile a grid environment worksheet that summarizes all the information required to set up your grid environment. A sample worksheet we used in our migration scenario is described in Table 2-2 on page 63.

Note: IBM offers a services engagement to help you design the optimal grid environment for your organization. Contact your local IBM sales representative for details on building an Information Platform & Solutions data integration grid.

1.7.2 Step 2: Building and testing your grid infrastructure using the BYOG toolkit

The BYOG toolkit provided (described in Section A.2, “Build Your Own Grid toolkit” on page 244) describes a step-by-step approach for building your grid environment.

1.7.3 Step 3: Reviewing all the jobs

Your existing InfoSphere Information Server has a number of DataStage, QualityStage, and Information Analyzer jobs.

Identifying and understanding all the jobs that must be migrated helps you to set certain upper limits:

- ▶ Grid environment global variable MaxNodesPerJob
- ▶ Job specific values such as COMPUTENODES and PARTITIONS.

1.7.4 Step 4: Installing and configuring the Grid Enablement Toolkit

The Grid Enablement Toolkit provided (described in “Grid Enablement Toolkit” on page 247) modifies the InfoSphere Information Server components to enable the creation of a dynamic configuration file. You must install and configure this to your grid environment.

1.7.5 Step 5: Tailoring existing DS/IA jobs to tolerate or exploit the grid environment

The Grid Enablement Toolkit describes the procedure for grid-enabling existing jobs (and repeated in 1.4.2, “How to grid-enable enterprise applications” on page 18).

Modify the existing jobs accordingly to tolerate or exploit the grid environment.

1.7.6 Step 6: Setting up the management of the grid environment

This involves setting up the management of the grid environment as follows:

- ▶ Administering and managing the resource manager such as Workload Scheduler LoadLeveler.
- ▶ Administering and managing the resource monitor such as Ganglia.
- ▶ Testing the failover and failback scenarios of your HA configuration. This is described in the migration scenario in “Step 4: Failing over to the standby node NILE” on page 221 and “Step 5: Failing back to the original front end node ORION” on page 236.
- ▶ Performance monitoring and tuning of the grid environment. This topic is beyond the scope of this Redbooks publication.

Note: IBM offers a services engagement to help you with administering and managing your grid environment. Contact your local IBM sales representative for details on contracting such Information Platform & Solutions services.

Migration scenario

In this chapter we describe a step-by-step approach to migrating an existing single server IBM Information Server environment on a Red Hat Enterprise Linux Advanced Server 4 platform to a grid environment. The grid environment involves one frontend (head) node, three compute nodes, and a dedicated node designated as the standby front end node for high availability.

The following topics are covered:

- ▶ Business requirements
- ▶ Environment configuration
- ▶ General approach
- ▶ Step 1: Installing and configuring the grid environment
- ▶ Step 2: Tailoring existing DS/IA jobs to operate in grid environment
- ▶ Step 3: Managing the grid environment
- ▶ Step 4: Failing over to the standby node NILE
- ▶ Step 5: Failing back to the original front end node ORION

2.1 Scenario Overview

Our scenario assumes a fictitious national department store named WantThatStuff that has built a star-schema-based sales analysis data warehouse with the following dimensions:

- ▶ Customer
- ▶ Store
- ▶ Product
- ▶ Date

Information Server on a Red Hat Enterprise Linux 4 platform was used to build and maintain this data warehouse. A daily update cycle was considered adequate for their business requirements.

Over time, the rapid expansion of WantThatStuff's business, with the addition of new stores and increased sales transaction volumes, began to impact the runtimes of the DataStage jobs. This affected WantThatStuff's ability to complete the daily update cycle within the available window for this activity.

2.2 Business requirements

The competitive nature of the retail industry required a cost-effective solution that addressed the daily batch window processing concerns while minimizing disruption to existing operations during migration. An added requirement was to implement a high availability (HA) environment for IBM Information Server, given its critical role in maintaining the star-schema data warehouse.

The two main alternatives were as follows:

- ▶ Upgrade the existing server hardware on which IBM Information Server is installed with more processors and memory.
- ▶ Retain the existing server, and deploy a grid environment by adding commodity hardware with 4 GB memory to the existing IBM Information Server platform.

From a viewpoint of cost effectiveness as well as minimizing disruption to existing operations, the grid deployment solution was considered the superior solution for their business requirements.

2.3 Environment configuration

The configuration of the planned grid environment is shown in Figure 2-1 on page 50. It shows one front end node, one dedicated standby front end node, and three compute nodes. For superior performance, a dedicated private network was chosen for the frontend and compute nodes interaction traffic.

We chose a dedicated standby front end node given the low cost nature of Intel®-based servers. Had our environment been IBM AIX®-based, we would almost certainly have opted for a standby node that performed processing unrelated to that performed on the primary front end node.

Important: Our decision to go with a four node configuration (excluding the dedicated standby node) was not based on capacity planning considerations, but on the fact that four nodes is a minimum grid configuration. This was acceptable because the focus of this Redbooks publication is on installing, configuring, and exploiting a grid environment involving IBM Information Server. Capacity planning considerations and performance issues are beyond the scope of this Redbooks publication.

You must determine the optimal number of compute nodes required for addressing your business requirements. This can be based on the number of concurrent jobs, number of head nodes accessing the grid concurrently, runtime requirements, and data volumes. All these factors must be taken into consideration when determining the number of compute nodes required.

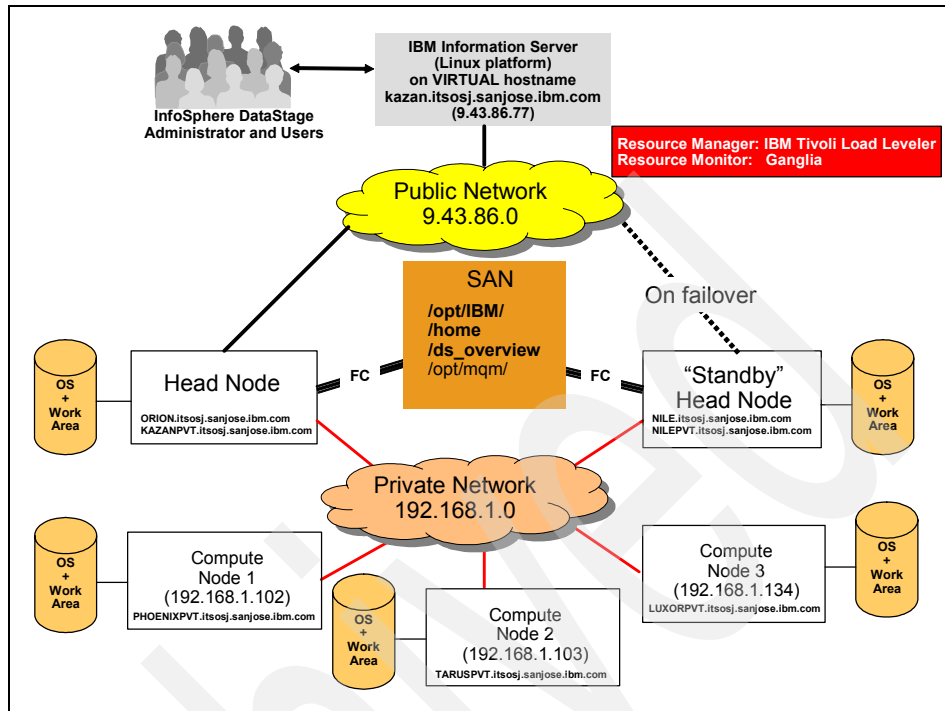


Figure 2-1 Grid environment configuration

Figure 2-1 shows the following:

- A Red Hat Enterprise Linux 4 server that has IBM Information Server and IBM WebSphere MQ installed. This is a requirement for the Distributed Transaction Stage that is used in updating WantThatStuff's star schema database.

Note: The Red Hat Enterprise Linux 4 server has a virtual host name of `kazan.itsosj.sanjose.ibm.com` and a virtual IP address of `9.43.86.77`. The actual host name is `ORION.itsosj.sanjose.ibm.com` with an IP address of `9.43.86.101`

This server has Dual Xeon Processors (2.80 GHz) with 8 GB of memory.

This server has two Network Interface Cards (NIC) as follows:

- First NIC is associated with host name `ORION.itsosj.sanjose.ibm.com` and the `9.43.86.0` network
- Second NIC is associated with host name `ORIONPVT.itsosj.sanjose.ibm.com` and the `192.168.1.0` network.

- ▶ One standby front end node that has two NICs as follows:
 - First NIC is associated with host name NILE.itsosj.sanjose.ibm.com and the 9.43.86.0 network
 - Second NIC is associated with host name NILEPVT.itsosj.sanjose.ibm.com and the 192.168.1.0 network.
- ▶ Three compute nodes with the following host names:
 - TARUSPVT.itsosj.sanjose.ibm.com
The TARUSPVT.itsosj.sanjose.ibm.com server has Dual Xeon Processors (2.80 GHz) with 8 GB of memory
 - LUXORPVT.itsosj.sanjose.ibm.com
The LUXORPVT.itsosj.sanjose.ibm.com server has six Dual Xeon Processors (1.95 GHz) with 6.92 GB of memory
 - PHOENIXPVT.itsosj.sanjose.ibm.com
The T PHOENIXPVT.itsosj.sanjose.ibm.com server has Dual Xeon Processors (2.80 GHz) with 8 GB of memory

All these host names are associated with the 192.168.1.0 network.

Each node has a local operating system, programs such as Workload Scheduler LoadLeveler and NIS clients, and work areas.

- ▶ A storage area network (SAN) that has four directories as follows:
 - The /opt/IBM/ directory contains the libraries for IBM Information Server.
 - The /home directory contains the user accounts information and libraries for DB2 UDB.
 - The /ds_overview directory contains the files that the DS_Overview project jobs use.
 - The /opt/mqm/ directory contains the libraries for IBM WebSphere MQ.

These directories are shared with the dedicated standby node NILE.itsosj.sanjose.ibm.com (in mutually exclusive fashion) because they are accessed on a failover.

- ▶ Two networks as follows:
 - The 9.43.86.0 “public” network (accessible through a Domain Name Server) is used by users and administrators to connect to the IBM Information Server from the Designer client and the Admin client.
 - The 192.168.1.0 “private” network (not accessible through a Domain Name Server) is used by the front end node to communicate with the compute nodes.
- ▶ Workload Scheduler LoadLeveler is the resource manager, and Ganglia is the resource monitor.

2.4 General approach

Figure 2-2 illustrates the general approach we adopted in executing the migration scenario.

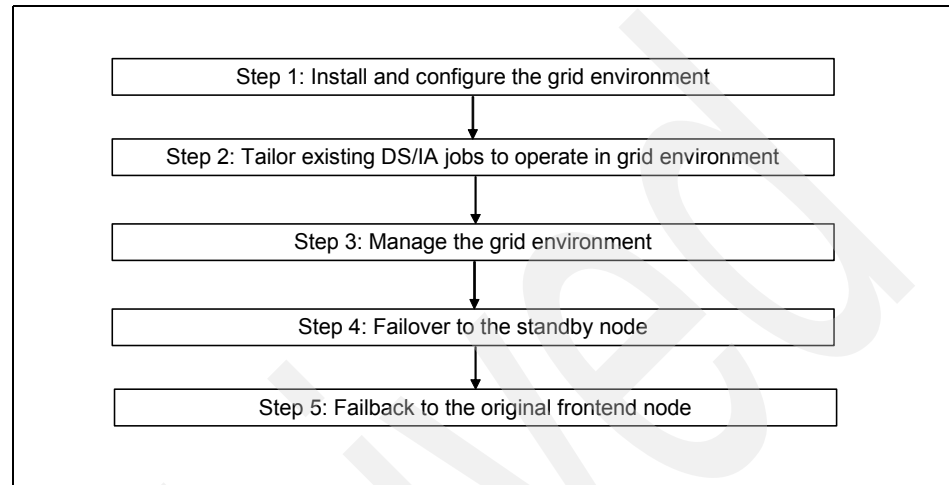


Figure 2-2 General approach

Attention: For HA purposes, we are configuring the virtual IP interfaces, IBM Information Server, and Grid (LoadLeveler and Ganglia) services on the primary front end node to be started manually after system boot, rather than automatically. This is to ensure that no conflicts arise after a successful failover to the standby node, when the original primary is rebooted while the standby node is in the role of the front end node.

Each of the steps shown in Figure 2-2 is briefly described here:

- ▶ **Step 1: Installing and configuring the grid environment**
This step describes the procedure for building the grid environment using the Build Your Own Grid (BYOG) toolkit, and configuring this grid environment using the Grid Enablement Toolkit. This step is described in detail in Section 2.5, “Step 1: Installing and configuring the grid environment” on page 54.
- ▶ **Step 2: Tailoring existing DS/IA jobs to operate in grid environment**
This step describes the procedure for migrating the existing DataStage and Information Analyzer jobs to the grid environment. This step is described in detail in Section 2.6, “Step 2: Tailoring existing DS/IA jobs to operate in grid environment” on page 146.

► Step 3: Managing the grid environment

This step describes procedures for managing the grid environment. This includes the following tasks:

- Viewing the status of the Workload Scheduler LoadLeveler queues
- Adding and removing compute nodes
- Monitoring the grid environment using Ganglia or other commands

This step is described in detail in Section 2.7, “Step 3: Managing the grid environment” on page 200.

► Step 4: Failing over to the standby node NILE

This step describes the procedure for failing over the IBM Information Server grid environment from the primary front end node to the standby node. This step is described in detail in Section 2.8, “Step 4: Failing over to the standby node NILE” on page 221.

► Step 5: Failing back to the original front end node ORION

This step describes the procedure for restoring the IBM Information Server grid environment from the standby front end node to the original front end node. This step is described in detail in Section 2.9, “Step 5: Failing back to the original front end node ORION” on page 236.

2.5 Step 1: Installing and configuring the grid environment

Step 1 describes the procedure for building the grid environment using the BYOG toolkit, and configuring this grid environment using the Grid Enablement Toolkit. This first step in executing the migration scenario consists of seven substeps. Figure 2-3 shows these substeps for installing and configuring the grid environment.

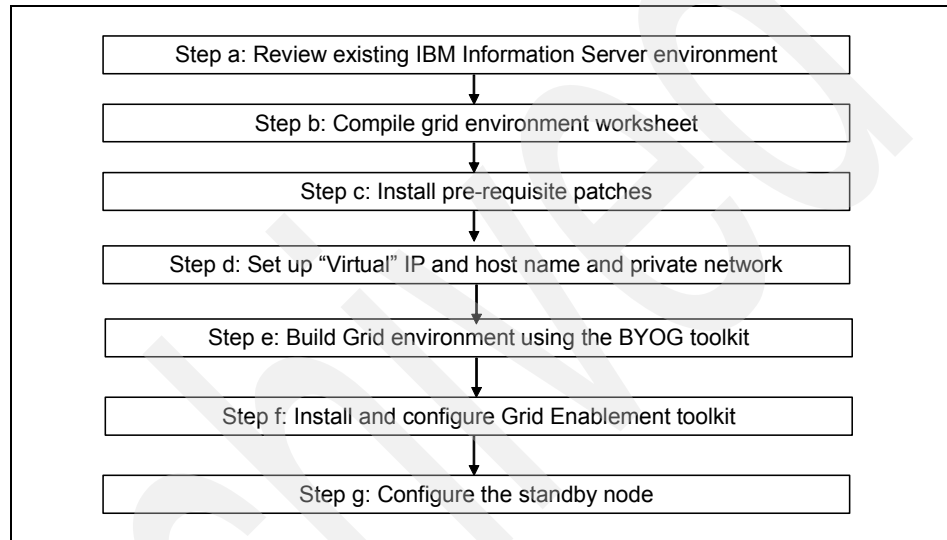


Figure 2-3 Steps in installing and configuring the grid environment

Each of the steps shown in Figure 2-3 is briefly described here:

- ▶ **Step 1a: Reviewing existing IBM Information Server environment**
This step is described in detail in Section 2.5.1, "Step 1a: Reviewing existing IBM Information Server environment" on page 55.
- ▶ **Step 1b: Compiling a grid environment worksheet**
Identify the information that you must compile prior to beginning the process of installing and configuring the grid environment. This step is described in detail in Section 2.5.2, "Step 1b: Compiling a grid environment worksheet" on page 63.
- ▶ **Step 1c: Installing pre-requisite patches**
Identify the pre-requisite patches and the actions to be taken after installing them. This step is described in detail in Section 2.5.3, "Step 1c: Installing pre-requisite patches" on page 66.

- ▶ **Step 1d: Setting up the Virtual IP and host name and private network**
Set up the virtual IP required for supporting the HA environment. It also includes the creation of a private network for the front end node and compute node interactions for superior performance. This step is described in detail in Section 2.5.4, “Step 1d: Setting up the Virtual IP and host name and private network” on page 68.
- ▶ **Step 1e: Building a Grid environment using the BYOG toolkit**
Build the designed grid environment infrastructure using the BYOG toolkit provided. This step is described in detail in Section 2.5.5, “Step 1e: Building a Grid environment using the BYOG toolkit” on page 78.
- ▶ **Step 1f: Installing and configuring Grid Enablement Toolkit**
Install and configure the Grid Enablement Toolkit that enables existing (pre-grid) DataStage, QualityStage, and Information Analyzer jobs to exploit the grid environment. This step is described in detail in Section 2.5.6, “Step 1f: Installing and configuring Grid Enablement Toolkit” on page 131.
- ▶ **Step 1g: Configuring the standby node**
Configure the standby node to permit it to takeover as the front end node when the original front end node fails. This step is described in detail in Section 2.5.7, “Step 1g: Configuring the standby node” on page 143.

2.5.1 Step 1a: Reviewing existing IBM Information Server environment

Review the existing IBM Information Server environment, including the IBM Information Server version, patches installed, NIC details, and directories to be shared. The information gathered is summarized in Table 2-1 on page 56.

Table 2-1 Main summary of the existing IBM Information Server environment

S#	Environment details	Values
1	IBM Information Server version fixpaks and patches	IBM Information Server v8.0.1.1 with the following fixpaks and patches: <ul style="list-style-type: none"> ▶ patch_e117664 ▶ patch_e122936 ▶ e121367 ▶ FixPack1.Redhat ▶ 120916g ▶ 12091612
2	NIC details <ul style="list-style-type: none"> ▶ Number available ▶ Number configured ▶ Number active 	<ul style="list-style-type: none"> ▶ 2 Ethernet ▶ 1 (eth0) with MAC Address 00:14:5E:B4:16:3A ▶ 1 (eth0) with MAC Address 00:14:5E:B4:16:3A
3	IP address	9.43.86.77
4	Subnet mask	255.255.252.0
5	Host Name	kazan.itsosj.sanjose.ibm.com
6	Gateway IP address	9.43.85.1
7	Nameserver IP address	9.43.85.11
8	Directories to be shared <ul style="list-style-type: none"> ▶ IBM Information Server home directory ▶ Home directory ▶ DataStage jobs data directory ▶ IBM WebSphere MQ directory ▶ IBM Tivoli Load Leveler directories 	<ul style="list-style-type: none"> ▶ /opt/IBM/ ▶ /home ▶ /ds_overview ▶ /opt/mqm/
9	/etc/hosts file contents	127.0.0.1 localhost.local domain localhost 9.43.86.77 kazan.itsosj.sanjose.ibm.com kazan
10	/etc/passwd file contents	contents not shown here
11	/etc/group file contents	contents not shown here
12	/etc/nsswitch file contents	passwd: files shadow: files group: files hosts: files dns netgroup: files

IBM Information Server version and patches

Version and patch information is available in the Version.xml file located in the /opt/IBM/InformationServer directory as shown in Example 2-1.

Example 2-1 /opt/IBM/InformationServer/ Version.xml file contents

```
<?xml version="1.0" encoding="UTF-8"?>
<ProductVersion specversion="1.0">
  <InstallType client="false" currentVersion="8.0.1.1" document="true"
domain="true" engine="true" repository="true"/>
  <Products>
    <Product installLocation="/opt/IBM/InformationServer/"
key="repositoryinstaller" name="repositoryinstaller"
version="8.0.1.0"/>
    <Product installLocation="/opt/IBM/InformationServer/"
key="wiiinstaller" name="wiiinstaller" version="8.0.1.0"/>
    <Product installLocation="/opt/IBM/InformationServer/"
key="servicesbackbone" name="servicesbackbone" version="8.0.1.0"/>
    <Product installLocation="/opt/IBM/InformationServer/"
key="datastagedocs" name="datastagedocs" version="8.0.1.0"/>
    <Product installLocation="/opt/IBM/InformationServer/"
key="datastage" name="datastage" version="8.0.1.0"/>
    <Product installLocation="/opt/IBM/InformationServer/"
key="businessglossary" name="businessglossary" version="8.0.1.0"/>
    <Product installLocation="/opt/IBM/InformationServer/"
key="informationserver" name="informationserver" version="8.0.1.0"/>
    <Product installLocation="/opt/IBM/InformationServer/"
key="informationanalyzer" name="informationanalyzer"
version="8.0.1.0"/>
  </Products>
  <History>
    <Sequence description="ibm_websphere_information_server" id="0"
installLocation="/opt/IBM/InformationServer" lastUpdateDate="" patch=""
rollback="" status="Success" version="8.0.1.0"/>
    <Sequence description="IA Rollup Patch 6. Summary ecase 120916.
Also includes non-IA components" id="1" installLocation=""
lastUpdateDate="Wed Jul 25 09:42:56 PDT 2007" patch="120916"
rollback="/opt/IBM/InformationServer/Updates/120916" status="Success"
version=""/>
    <Sequence description="Rollup Patch 6. Summary ecase 120916" id="2"
installLocation="" lastUpdateDate="Sat Sep 01 00:56:40 PDT 2007"
patch="120916g" rollback="/opt/IBM/InformationServer/Updates/120916g"
status="Success" version=""/>
    <Sequence description="Information Server 8.0.1 Fix Pack 1" id="3"
installLocation="" lastUpdateDate="Mon Nov 26 17:33:05 PST 2007"
```

```

patch="FixPack1.Redhat"
rollback="/opt/IBM/InformationServer/Updates/FixPack1.Redhat"
status="Success" version=""/>
  <Sequence description="Component Installer patch for ECASE 121367
-- supports DTStage new properties" id="4" installLocation=""
lastUpdateDate="Mon Nov 26 17:37:16 PST 2007" patch="e121367"
rollback="/opt/IBM/InformationServer/Updates/e121367" status="Success"
version=""/>
  <Sequence description="Server side components for deploying the
Distributed Transaction Stage" id="5" installLocation=""
lastUpdateDate="Mon Nov 26 17:45:06 PST 2007" patch="patch_e122936"
rollback="/opt/IBM/InformationServer/Updates/patch_e122936"
status="Success" version=""/>
  <Sequence description=" XML Reader and XML Writer library patch for
the ecase 117664" id="6" installLocation="" lastUpdateDate="Mon Nov 26
18:09:07 PST 2007" patch="patch_e117664"
rollback="/opt/IBM/InformationServer/Updates/patch_e117664"
status="Success" version=""/>
</History>
<Properties>
  <Property name="DB2InstallLocation" value="/opt/IBM/db2/V9"/>
  <Property name="AsbPortNumber" value="9080"/>
  <Property name="AppServerUserName" value="wasadmin"/>
  <Property name="DB2InstanceOwnerName" value="db2inst1"/>
  <Property name="DB2PortNumber" value="50001"/>
  <Property name="AppServerHome"
value="/opt/IBM/WebSphere/AppServer"/>
  <Property name="IADatabaseUserName" value="iauser"/>
  <Property name="VPD"
value="/root/InstallShield/Universal/IBM/InformationServer/Gen2/_vpddb/
vpd.script"/>
  <Property name="XMetaDatabaseName" value="xmeta"/>
  <Property name="DB2CopyName" value=""/>
  <Property name="AppServerName" value="websphere"/>
  <Property name="DSEngineLocation"
value="/opt/IBM/InformationServer/Server/DSEngine"/>
  <Property name="InstallLocation"
value="/opt/IBM/InformationServer"/>
  <Property name="XMetaDatabaseUserName" value="xmeta"/>
  <Property name="XMetaDatabasePortNumber" value=""/>
  <Property name="IADatabaseName" value="iadb"/>
  <Property name="AsbHostName" value="Kazan.itsosj.sanjose.ibm.com"/>
  <Property name="AsbAgentHostName"
value="Kazan.itsosj.sanjose.ibm.com"/>

```



```
<Property name="WebSphereProfileLocation"
value="/opt/IBM/WebSphere/AppServer/profiles/default"/>
</Properties>
</ProductVersion>
```

NIC details and IP address

The list of Ethernet Network Interface Cards (NIC) installed on the server, their characteristics, and their status can be determined by issuing the following commands:

► **lspci**

Example 2-2 shows the results of the **lspci** command filtered to show only Ethernet adapters. It shows the existence of two Broadcom Corporation Ethernet adapters.

Example 2-2 lspci command output

```
[root@kazan ~]# lspci | grep Ethernet
06:00.0 Ethernet controller: Broadcom Corporation NetXtreme BCM5721
Gigabit Ethernet PCI Express (rev 11)
07:00.0 Ethernet controller: Broadcom Corporation NetXtreme BCM5721
Gigabit Ethernet PCI Express (rev 11)
```

► **service network status**

Example 2-3 shows the results of the **service network status** command.

It shows one configured device, **eth0**, which also happens to be active. This means that only one of the two Ethernet adapters identified in Example 2-2 has been configured.

Example 2-3 Service network status command output

```
[root@kazan ~]# service network status
Configured devices:
lo eth0
Currently active devices:
lo eth0
```

Note: When you have configured two interfaces (eth0 and eth1) and both are active, the **service network status** command shows the following output:

```
[root@orion ~]# service network status
Configured devices:
lo eth0 eth1
Currently active devices:
lo eth0 eth1
```

Next, if you and bring down one of the interfaces using the **ifdown** command (**ifdown eth1**), the **service network status** command shows the following output:

```
[root@orion ~]# service network status
Configured devices:
lo eth0 eth1
Currently active devices:
lo eth0
```

► **ifconfig**

Example 2-4 shows results of issuing the **ifconfig** command that provides details about the IP address (9.43.86.77), MAC Address (HWaddr 00:14:5E:B4:16:3A), and subnet mask (Mask:255.255.252.0) associated with each interface.

Example 2-4 ifconfig command output

```
[root@kazan ~]# ifconfig
eth0      Link encap:Ethernet  HWaddr 00:14:5E:B4:16:3A
          inet addr:9.43.86.77  Bcast:9.43.87.255  Mask:255.255.252.0
          inet6 addr: fe80::214:5eff:feb4:163a/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:976111 errors:0 dropped:0 overruns:0 frame:0
          TX packets:445119 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:201382012 (192.0 MiB)  TX bytes:169549021 (161.6
MiB)

          Interrupt:169
```

Host Name and gateway IP address

The host name and gateway IP address are stored in the network file in the /etc/sysconfig directory as shown in Example 2-5.

Example 2-5 Host Name and gateway address

```
[root@kazan ~]# more /etc/sysconfig/network
NETWORKING=yes
HOSTNAME=kazan.itsosj.sanjose.ibm.com
GATEWAY=9.43.85.1
```

Nameserver IP address

The nameserver IP address is stored in the resolv.conf file in the /etc directory as shown in Example 2-6.

Example 2-6 Nameserver IP address

```
[root@kazan ~]# more /etc/resolv.conf
search itsosj.sanjose.ibm.com
nameserver 9.43.85.11
```

Directories to be shared

The directories to be shared between the front end node and the standby node are as follows:

- ▶ Information Server home directory
/opt/IBM/
- ▶ DB2 home and user accounts directory
/home/
- ▶ DataStage jobs directory
/ds_overview
- ▶ IBM WebSphere MQ home directory
/opt/mqm/

Some of these directories are also accessed by the compute nodes (more on this later).

/etc/hosts file contents

The hosts file maps IP addresses to machine or domain names and an optional nickname. See Example 2-7 on page 62. This file is typically consulted first for name resolution.

Example 2-7 /etc/hosts file contents

```
[root@kazan ~]# more /etc/hosts
# Do not remove the following line, or various programs
# that require network functionality will fail.
127.0.0.1                localhost.localdomain localhost
9.43.86.77               kazan.itsosj.sanjose.ibm.com kazan
```

/etc/passwd file contents

The passwd file stores user account information required for login, and contains one entry for each user of the system. A partial list is shown in Example 2-8.

Example 2-8 /etc/passwd file contents

```
[root@kazan ~]# more /etc/passwd
root:x:0:0:root:/root:/bin/bash
bin:x:1:1:bin:/bin:/sbin/nologin
daemon:x:2:2:daemon:/sbin:/sbin/nologin
adm:x:3:4:adm:/var/adm:/sbin/nologin
.....
xmeta:x:501:501::/home/xmeta:/bin/bash
dsadm:x:507:512::/home/dsadm:/bin/bash
.....
```

/etc/group file contents

The group file defines the groups to which users belong. Each entry identifies the name of a group (db2inst1) and members of the group (root, admin, dsadm, and nagraj). A partial list is shown in Example 2-9.

Example 2-9 /etc/group file contents

```
[root@kazan ~]# more /etc/group
root:x:0:root
bin:x:1:root,bin,daemon
daemon:x:2:root,bin,daemon
sys:x:3:root,bin,adm
adm:x:4:root,adm,daemon
.....
xmeta:x:501:xmeta,db2inst1
dstage:x:512:dsadm,admin
.....
```

/etc/nsswitch.conf file contents

The /etc/nsswitch.conf file specifies the services to be used when determining information such as passwords, shadow, groups, hosts, and netgroup files. Example 2-10 shows the contents indicating that such information must be obtained from the files. In the case of hosts, the /etc/hosts file must be consulted first before referencing the DNS.

Example 2-10 /etc/nsswitch.conf file contents

```
[root@kazan ~]# more /etc/nsswitch.conf | egrep 'passwd|shadow|group|hosts'
passwd:    files
shadow:    files
group:     files
hosts:     files dns
netgroup:  files
```

2.5.2 Step 1b: Compiling a grid environment worksheet

To ensure a smooth and successful installation and configuration of your planned grid environment, we strongly recommend that you first compile the information shown in the worksheet in Table 2-2. This worksheet shows the information collected for our planned grid environment configuration shown in Figure 2-1 on page 50.

Table 2-2 Worksheet for grid environment

S#	Environment details	Values
1	Front end node host name	ORION.itsosj.sanjose.ibm.com
2	Front end node "public" real IP address	9.43.86.101
3	Front end node "public" real IP address domain name	ORION.itsosj.sanjose.ibm.com
4	Front end node "public" virtual IP address	9.43.86.77
5	Front end node "public" virtual IP address domain name	kazan.itsosj.sanjose.ibm.com
6	Front end node "private" real IP address	192.168.1.101
7	Front end node "private" real IP address domain name	ORIONPVT.itsosj.sanjose.ibm.com
8	Front end node "private" virtual IP address	192.168.1.77
9	Front end node "private" virtual IP address domain name	KAZANPVT.itsosj.sanjose.ibm.com

S#	Environment details	Values
10	Front end node gateway IP address & subnet mask	9.43.85.1 and 255.255.252.0
11	Standby node host name	NILE.itsosj.sanjose.ibm.com
12	Standby node “public” real IP address	9.43.86.76
13	Standby node “public” real IP address domain name	NILE.itsosj.sanjose.ibm.com
14	Standby node “private” real IP address	192.168.1.76
15	Standby node “private” real IP address domain name	NILEPVT.itsosj.sanjose.ibm.com
16	Compute node information <ul style="list-style-type: none"> ▶ Compute node 1 <ul style="list-style-type: none"> – “private” real IP address – “private” real IP address domain name – MAC Address eth0 – MAC Address eth1 ▶ Compute node 2 <ul style="list-style-type: none"> – “private” real IP address – “private” real IP address domain name – MAC Address eth0 – MAC Address eth1 ▶ Compute node 3 <ul style="list-style-type: none"> – “private” real IP address – “private” real IP address domain name – MAC Address eth0 – MAC Address eth1 	<ul style="list-style-type: none"> ▶ Compute node 1 <ul style="list-style-type: none"> – 192.168.1.102 – PHOENIXPVT.itsosj.sanjose.ibm.com – 00:14:5E:AC:36:33 – 00:14:5E:AC:36:32 ▶ Compute node 2 <ul style="list-style-type: none"> – 192.168.1.103 – TARUSPVT.itsosj.sanjose.ibm.com – 00:14:5E:A4:E4:C6 – 00:14:5E:A4:E4:C7 ▶ Compute node 3 <ul style="list-style-type: none"> – 192.168.1.134 – LUXORPVT.itsosj.sanjose.ibm.com – 00:03:47:13:D7:F1 – 00:02:55:AC:86:D2
17	NISDomain ¹ name	IS.redbook.grid
18	Mount directories on the compute nodes	/opt/IBM/ /home/ /ds_overview/
19	Shared directories on the standby node	/opt/IBM/ /home/ /ds_overview/ /opt/mqm/

S#	Environment details	Values
20	DHCP server information <ul style="list-style-type: none"> ▶ IP range dynamic-bootp (for PXE boot) ▶ Filename ▶ Operating system location on 192.168.1.101 	<ul style="list-style-type: none"> ▶ 192.168.1.102 to 192.168.1.103 ▶ /linux-install/prelinux.0 ▶ /tftpboot/Redhat4
21	<p>Network.# files² for each of the compute nodes</p> <ul style="list-style-type: none"> ▶ network.102 ▶ network.103 ▶ network.134 	<ul style="list-style-type: none"> ▶ network --device eth1 --bootproto static --ip 192.168.1.102 --netmask 255.255.252.0 --gateway 9.43.85.1 --nameserver 9.43.85.11 --hostname phoenixpvt.itsosj.sanjose.ibm.com network --device eth0 --bootproto static --ip 9.43.86.102 --netmask 255.255.252.0 --gateway 9.43.85.1 --nameserver 9.43.85.11 --hostname phoenix.itsosj.sanjose.ibm.com ▶ network --device eth1 --bootproto static --ip 192.168.1.103 --netmask 255.255.252.0 --gateway 9.43.85.1 --nameserver 9.43.85.11 --hostname taruspvt.itsosj.sanjose.ibm.com network --device eth0 --bootproto static --ip 9.43.86.103 --netmask 255.255.252.0 --gateway 9.43.85.1 --nameserver 9.43.85.11 --hostname tarus.itsosj.sanjose.ibm.com ▶ network --device eth1 --bootproto static --ip 192.168.1.134 --netmask 255.255.252.0 --gateway 9.43.85.1 --nameserver 9.43.85.11 --hostname luxorpvt.itsosj.sanjose.ibm.com network --device eth0 --bootproto static --ip 9.43.86.134 --netmask 255.255.252.0 --gateway 9.43.85.1 --nameserver 9.43.85.11 --hostname luxor.itsosj.sanjose.ibm.com
<p>¹ An NIS domainname is the name for a group of related hosts.</p> <p>² This file provides network information such as host name, IP address, subnet mask, gateway IP address, and nameserver information</p>		

2.5.3 Step 1c: Installing pre-requisite patches

The following patches are required for supporting the grid environment on IBM Information Server Version 8.0.1:

Note: Neither of these patches currently exist on our IBM Information Server as identified in Example 2-1 on page 57.

► patch_e126149v4

This patch fixes a problem where status records for a job were being removed in the window where a job has started but has not written any log entries.

Note: DataStage must be up and running when installing this patch.

► patch_e124940

Without this patch, Table Definition locators and Operational Metadata uses the physical host name of the server on which IBM Information Server is installed, rather than a logical server.

Note: The real host name, as reported by the `hostname` or `gethostname()` commands, is used for namespace qualification of persistent metadata.

This is a problem in an HA solution environment when the takeover node has a different physical host name. This results in DataStage projects not working when failed over to the standby node.

This patch defines a new environment variable `DS_HOSTNAME_ALIAS` and modifies several modules to use this value to override the physical host name. This allows the real host name to be different in an HA environment involving a standby node, and for everything to continue working after a failover if the `DS_HOSTNAME_ALIAS` environment variable always has the same logical host name.

The `DS_HOSTNAME_ALIAS` environment variable is assigned the name `kazan.itsosj.sanjose.ibm.com` that is the same as the current physical host name. We will change `kazan.itsosj.sanjose.ibm.com` to a logical (or virtual) host name and assign a new physical host name (`ORION.itsosj.sanjose.ibm.com`) in Section 2.5.4, “Step 1d: Setting up the Virtual IP and host name and private network” on page 68 to support our HA solution.

After this patch has been installed, you must perform the following steps:

- a. Modify the dsenv file with the new environment variable as shown in Example 2-11.

Example 2-11 Modify dsenv file

```
.....  
### For Patch ECASE: 124940  
DS_HOSTNAME_ALIAS=kazan.itsosj.sanjose.ibm.com  
export DS_HOSTNAME_ALIAS
```

- b. Source the dsenv file as shown in Example 2-12.

Example 2-12 Source the dsenv file as root

```
. $DSHOME/dsenv
```

- c. For the changes to take effect, perform the following steps:
 - i. Check that no clients or jobs are running (as shown in Example 2-13).
 - ii. Stop the instance (as shown in Example 2-14 on page 68).
 - iii. Start the instance (as shown in Example 2-15 on page 68).

Example 2-13 Check that no clients or jobs are running

```
$DSHOME/bin/uv -admin -info
```

```
Details for DataStage Engine release 8.0.1.0 instance "ade"  
=====
```

Install history	: Installed by root (admin:dsadm) on: Tue Jul 24 18:44:04 PDT 2007
Instance tag	: ade
Engine status	: Running
Engine location	: /opt/IBM/InformationServer/Server/DSEngine
Binary location	: /opt/IBM/InformationServer/Server/DSEngine/bin
Impersonation	: Enabled
Administrator	: dsadm
Autostart mode	: enabled
Autostart link(s)	: /etc/rc.d/init.d/ds.rc : /etc/rc.d/rc2.d/S999ds.rc : /etc/rc.d/rc3.d/S999ds.rc : /etc/rc.d/rc4.d/S999ds.rc : /etc/rc.d/rc5.d/S999ds.rc
Startup script	: /opt/IBM/InformationServer/Server/DSEngine/sample/ds.rc

```
Cache Segments      :0 active
User Segments       :0 active

DSnum Uid      Pid  Ppid  C Stime Tty      Time      Command
```

Example 2-14 Stop the instance

```
$DSHOME/bin/uv -admin -stop
```

Example 2-15 Start the instance

```
$DSHOME/bin/uv -admin -start
```

- d. Verify that the environment variable changes have taken effect as shown in Example 2-16.

Example 2-16 Verify that the environment variables have taken effect

```
#Get the PID for the DataStage Engine
ps -ef | grep dsr | grep -v grep
      root      25059      1  0 09:31 pts/4      00:00:00
/opt/IBM/InformationServer/Server/DSEngine/bin/dsrpcd
# Get the environment variables for the running PID 25059
(cat /proc/25059/environ ; echo) | tr "\000" "\n" | grep -i hostname
DS_HOSTNAME_ALIAS=kazan.itsosj.sanjose.ibm.com
```

2.5.4 Step 1d: Setting up the Virtual IP and host name and private network

Set up the Virtual IP on the existing server to support the HA environment and verify that existing jobs continue to work. This step also details the creation of a required private network for the front end node and compute node interactions for superior performance.

Attention: The following steps were necessary in our environment because we manually set up our HA environment. In practice, organizations use HA software such as Heartbeat to perform these tasks for you.

Heartbeat is available from the from the Linux-HA Project at the following Web page:

<http://www.linux-ha.org>

To Set up Virtual IP and host name and private network, perform the following steps:

1. Set up the Virtual IP address.

You must create or assign a new real IP address and then convert the existing IP address to the Virtual IP address as follows:

- a. Modify the IP address in the ifcfg-eth0 interface configuration file in the /etc/sysconfig/network-scripts directory from a value of 9.43.86.77 to a value of 9.43.86.101.

The original IP address (9.43.86.77) becomes the Virtual IP address, and the new value (9.43.86.101) becomes the real IP address. The modified contents are shown in Example 2-17.

Note: The ONBOOT=yes option specifies that this device must be activated at boot-time. This option only applies to real IP addresses and not to virtual IP addresses.

Example 2-17 /etc/sysconfig/network-scripts/ifcfg-eth0 file: Modified contents

```
DEVICE=eth0
BOOTPROTO=static
HWADDR=00:14:5E:B4:16:3A
IPADDR=9.43.86.101
NETMASK=255.255.252.0
ONBOOT=yes
TYPE=Ethernet
```

- b. Create a new alias interface configuration file ifcfg-eth0:1 in the /etc/sysconfig/network-scripts directory. This corresponds to the Virtual IP address definition associated with the eth0 device as shown in Example 2-18 on page 70. The Virtual IP address is 9.43.86.77 (previously the real IP address).

Note: The ONPARENT=no option specifies that this device must not be activated at boot-time. The default is yes. As mentioned earlier, the ONBOOT option only applies to real IP addresses, and the ONPARENT option performs the corresponding function for virtual IP addresses. The parent to the virtual IP addresses is the real IP address corresponding to it.

With the ONPARENT=no option, this virtual IP interface does not start up on boot. You must therefore explicitly start it with the **ifup eth0:1** command.

Example 2-18 /etc/sysconfig/network-scripts/ifcfg-eth0:1 file contents

```
DEVICE=eth0:1
BOOTPROTO=static
HWADDR=00:14:5E:B4:16:3A
IPADDR=9.43.86.77
NETMASK=255.255.252.0
ONPARENT=no
TYPE=Ethernet
```

2. Set up the private network.

Create the following files in the /etc/sysconfig/network-scripts directory corresponding to the second Ethernet adapter (MAC Address 00:14:5E:B4:16:3B):

– ifcfg-eth1 interface configuration file

This file corresponds to the private network (192.168.1.0) with the real IP address of 192.168.1.101 as shown in Example 2-19.

Example 2-19 /etc/sysconfig/network-scripts/ifcfg-eth1 file contents

```
DEVICE=eth1
BOOTPROTO=static
HWADDR=00:14:5E:B4:16:3B
IPADDR=192.168.1.101
NETMASK=255.255.252.0
ONBOOT=yes
TYPE=Ethernet
```

– ifcfg-eth1:1 alias interface configuration file

This file corresponds to the private network (192.168.1.0) with the Virtual IP address of 192.168.1.77 as shown in Example 2-20 on page 71.

Note: The ONPARENT=no option specifies that this device must not be activated at boot-time. The default is yes. The ONBOOT option only applies to real IP addresses, and the ONPARENT option performs the corresponding function for virtual IP addresses. The parent to the virtual IP addresses is the real IP address corresponding to it.

With the ONPARENT=no option, this virtual IP interface does not start up on boot. You must start it with the **ifup eth1:1** command.

Example 2-20 /etc/sysconfig/network-scripts/ifcfg-eth1:1 file contents

```
DEVICE=eth1:1
BOOTPROTO=static
HWADDR=00:14:5E:B4:16:3B
IPADDR=192.168.1.77
NETMASK=255.255.252.0
ONPARENT=no
TYPE=Ethernet
```

After the interface configuration files have been updated or created, reboot the server for the changes to take effect.

After reboot, perform the following steps:

1. Start the virtual IP interfaces eth0:1 and eth1:1 using the **ifup** commands.
2. Start the IBM Information Server using the appropriate commands
3. Associate the newly added IP addresses with domain names by updating the `/etc/hosts` file as shown in Example 2-21.

The new public address (9.43.86.101) and private addresses (192.168.1.77 and 192.168.1.101) and their corresponding domain names and nicknames are highlighted.

Example 2-21 /etc/hosts file: Modified contents

```
# Do not remove the following line, or various programs
# that require network functionality will fail.
127.0.0.1          localhost.localdomain localhost
### Public Addresses
9.43.86.77         kazan.itsosj.sanjose.ibm.com kazan
9.43.86.101        orion.itsosj.sanjose.ibm.com  orion
### Private Addresses
192.168.1.77       kazanpvt.itsosj.sanjose.ibm.com kazanpvt
192.168.1.101      orionpvt.itsosj.sanjose.ibm.com orionpvt
```

4. Set the host name to the domain name (ORION.itsosj.sanjose.ibm.com) on the public network by modifying the network file in the `/etc/sysconfig` directory as shown in Example 2-22 on page 72.

Note: The physical host name that you define here must match the Workload Scheduler LoadLeveler central manager name specified when you initialize loadl on front end node (more on this later).

This step ensures that you explicitly define what it must be. In our case, it is the domain name ORION.itsosj.sanjose.ibm.com associated with the new public address 9.43.86.101

Example 2-22 Set the host name in the network file in the /etc/sysconfig directory

```
NETWORKING=yes
HOSTNAME=orion.itsosj.sanjose.ibm.com
GATEWAY=9.43.85.1
```

5. Verify the host name is set correctly by issuing the **hostname** command as shown in Example 2-23.

Example 2-23 Verify the host name

```
[root@orion IBM]# hostname
orion.itsosj.sanjose.ibm.com
```

6. Verify that all the devices defined earlier have started successfully, and that the domain names can be pinged successfully.

The **ifconfig** command output (Example 2-24) shows the eth0, eth0:1, eth1, and eth1:1 devices with their corresponding IP addresses.

Example 2-24 ifconfig command output

```
[root@orion IBM]# ifconfig
eth0      Link encap:Ethernet  HWaddr 00:14:5E:B4:16:3A
          inet addr:9.43.86.101  Bcast:9.43.87.255  Mask:255.255.252.0
          inet6 addr: fe80::214:5eff:feb4:163a/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:1339521 errors:0 dropped:0 overruns:0 frame:0
          TX packets:766449 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:183792784 (175.2 MiB)  TX bytes:271346984 (258.7
MiB)
          Interrupt:169

eth0:1    Link encap:Ethernet  HWaddr 00:14:5E:B4:16:3A
          inet addr:9.43.86.77  Bcast:9.43.87.255  Mask:255.255.252.0
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          Interrupt:169
```

```

eth1      Link encap:Ethernet  HWaddr 00:14:5E:B4:16:3B
          inet addr:192.168.1.101  Bcast:192.168.3.255
Mask:255.255.252.0
          inet6 addr: fe80::214:5eff:feb4:163b/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:104900 errors:0 dropped:0 overruns:0 frame:0
          TX packets:1227864 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:13957106 (13.3 MiB)  TX bytes:1797068640 (1.6 GiB)
          Interrupt:169

eth1:1    Link encap:Ethernet  HWaddr 00:14:5E:B4:16:3B
          inet addr:192.168.1.77  Bcast:192.168.3.255
Mask:255.255.252.0
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          Interrupt:169

lo        Link encap:Local Loopback
          inet addr:127.0.0.1  Mask:255.0.0.0
          inet6 addr: ::1/128 Scope:Host
          UP LOOPBACK RUNNING  MTU:16436  Metric:1
          RX packets:2805938 errors:0 dropped:0 overruns:0 frame:0
          TX packets:2805938 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:0
          RX bytes:929250275 (886.2 MiB)  TX bytes:929250275 (886.2
MiB)

```

Example 2-25 through Example 2-28 on page 74 show the successful pinging of the domain names orion, orionpvt, kazanpvt, and kazan.

Example 2-25 ping orion

```

[root@orion ~]# ping -c 1 orion
PING orion.itsosj.sanjose.ibm.com (9.43.86.101) 56(84) bytes of data.
64 bytes from orion.itsosj.sanjose.ibm.com (9.43.86.101): icmp_seq=0 ttl=64
time=0.056 ms

--- orion.itsosj.sanjose.ibm.com ping statistics ---
1 packets transmitted, 1 received, 0% packet loss, time 0ms
rtt min/avg/max/mdev = 0.056/0.056/0.056/0.000 ms, pipe 2

```

Example 2-26 ping orionpvt

```
[root@orion pxeboot]# ping -c 1 orionpvt
PING orionpvt.itsosj.sanjose.ibm.com (192.168.1.101) 56(84) bytes of data.
64 bytes from orionpvt.itsosj.sanjose.ibm.com (192.168.1.101): icmp_seq=0
ttl=64 time=0.037 ms
```

```
--- orionpvt.itsosj.sanjose.ibm.com ping statistics ---
1 packets transmitted, 1 received, 0% packet loss, time 0ms
rtt min/avg/max/mdev = 0.037/0.037/0.037/0.000 ms, pipe 2
```

Example 2-27 ping kazanpvt

```
[root@orion pxeboot]# ping -c 1 kazanpvt
PING kazanpvt.itsosj.sanjose.ibm.com (192.168.1.77) 56(84) bytes of data.
64 bytes from kazanpvt.itsosj.sanjose.ibm.com (192.168.1.77): icmp_seq=0 ttl=64
time=2.92 ms
```

```
--- kazanpvt.itsosj.sanjose.ibm.com ping statistics ---
1 packets transmitted, 1 received, 0% packet loss, time 0ms
rtt min/avg/max/mdev = 2.925/2.925/2.925/0.000 ms, pipe 2
```

Example 2-28 ping kazan

```
[root@orion pxeboot]# ping -c 1 kazan
PING kazan (9.43.86.77) 56(84) bytes of data.
64 bytes from kazan (9.43.86.77): icmp_seq=0 ttl=64 time=0.190 ms
```

```
--- kazan ping statistics ---
1 packets transmitted, 1 received, 0% packet loss, time 0ms
rtt min/avg/max/mdev = 0.190/0.190/0.190/0.000 ms, pipe 2
```

7. Because we want interactions between the conductor node and the compute nodes to occur on the private network, perform the following updates to the dsenv file and the default.apt configuration file:
 - a. The \$APT_PM_CONDUCTOR_HOSTNAME environment variable determines the node name that section leaders on the compute node use to connect back to the conductor. Refer to the Parallel Framework in Appendix C, “Parallel Framework” on page 303. Specify the node name corresponding to the private network (kazanpvt.itsosj.sanjose.ibm.com), which must then be exported as shown in Example 2-29 on page 75.

Note: These changes are dynamic and do not require a reboot.

Example 2-29 Modify dsenv file

```
.....  
### For virtual host name being kazanpvt.itsosj.sanjose.ibm.com  
APT_PM_CONDUCTOR_HOSTNAME=kazanpvt.itsosj.sanjose.ibm.com  
export APT_PM_CONDUCTOR_HOSTNAME  
.....
```

- b. The default.apt configuration file must also be modified so that the fastnames corresponding to the conductor and compute nodes referenced the private network as shown in Example 2-30 where kazan.itsosj.sanjose.ibm.com (public network) was replaced with kazanpvt.itsosj.sanjose.ibm.com (private network) as highlighted.

Example 2-30 default.apt file: Modified contents

```
{  
  node "node1"  
  {  
    fastname "kazanpvt.itsosj.sanjose.ibm.com"  
    pools "conductor"  
    resource disk "/opt/IBM/InformationServer/Server/Datasets" {pools ""}  
    resource scratchdisk "/opt/IBM/InformationServer/Server/Scratch" {pools  
""}  
  }  
  node "node2"  
  {  
    fastname "kazanpvt.itsosj.sanjose.ibm.com"  
    pools ""  
    resource disk "/opt/IBM/InformationServer/Server/Datasets" {pools ""}  
    resource scratchdisk "/opt/IBM/InformationServer/Server/Scratch" {pools  
""}  
  }  
  node "node3"  
  {  
    fastname "kazanpvt.itsosj.sanjose.ibm.com"  
    pools ""  
    resource disk "/opt/IBM/InformationServer/Server/Datasets" {pools ""}  
    resource scratchdisk "/opt/IBM/InformationServer/Server/Scratch" {pools  
""}  
  }  
}
```

8. Verify the success of all the changes made so far, by running an existing job, as shown in Figure 2-4 through Figure 2-7 on page 77.

The results of this execution are shown in Figure 2-5. The DataStage and QualityStage Director log (Figure 2-6 on page 77) shows the progress of the execution identifying the default.apt configuration file being used. The default.apt configuration file contents are shown in Figure 2-7 on page 77.

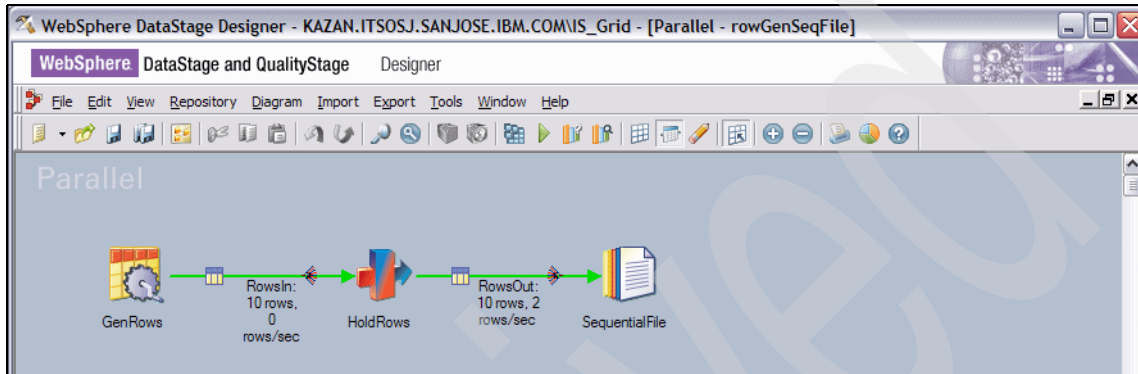


Figure 2-4 rowGenSleeper with sequential file, part 1 of 4

The Data Browser window displays the output of the 'SequentialFile.RowsOut' stage. It shows a table with three columns: c1, c2, and c3. The data is as follows:

c1	c2	c3
aaaaaaaaa	1000000	aaaaaaaaaaaaaaaaa
cccccccc	1000002	c
eeeeeeee	1000004	eeeeee
gggggggg	1000006	gggggg
iiiiiiii	1000008	iiiiiiiiiiiiii
bbbbbbbb	1000001	bbbbbb
dddddddd	1000003	dddddddddddddd
fffffffe	1000005	fffffffeffff
hhhhhhhh	1000007	hhhhhhh
jjjjjjjj	1000009	jjjjjj

Figure 2-5 rowGenSleeper with sequential file, part 2 of 4

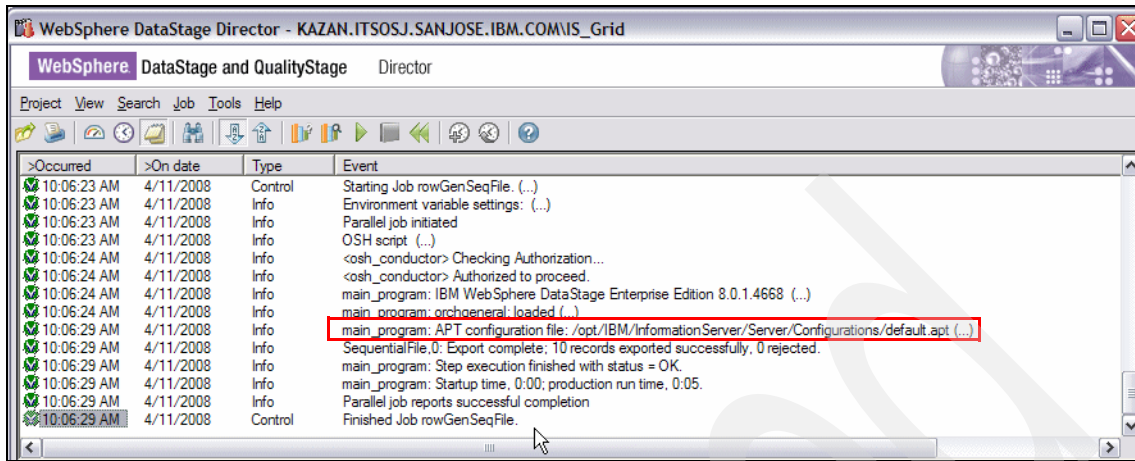


Figure 2-6 rowGenSleeper with sequential file, part 3 of 4

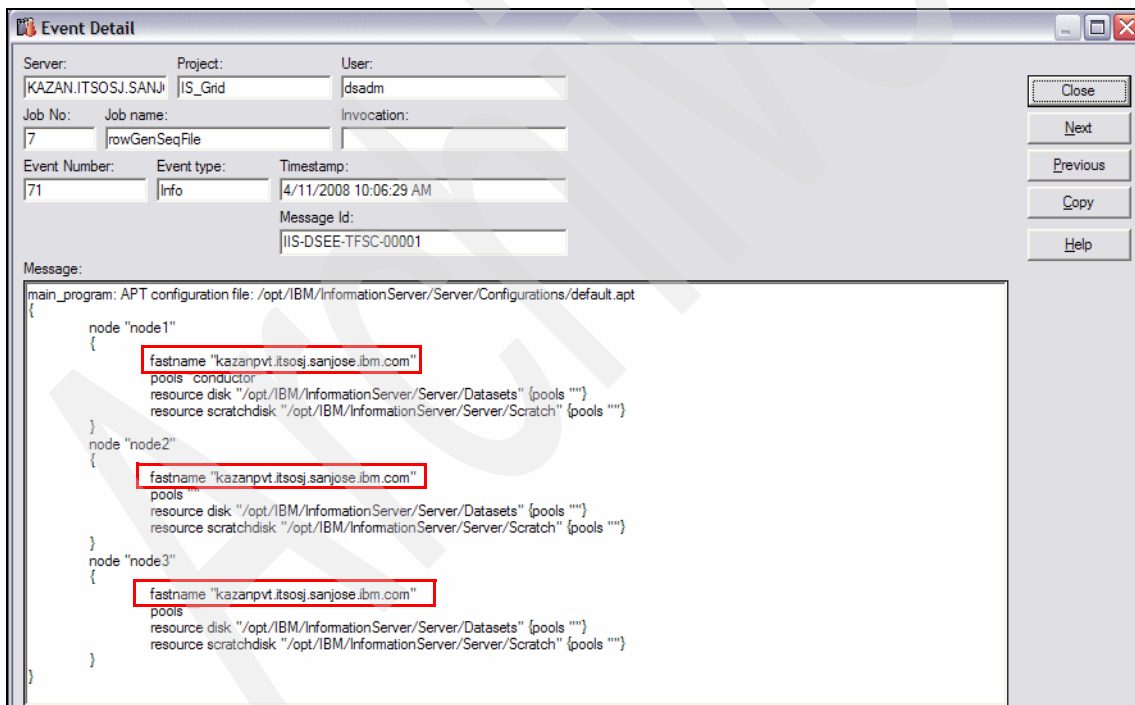


Figure 2-7 rowGenSleeper with sequential file, part 4 of 4

2.5.5 Step 1e: Building a Grid environment using the BYOG toolkit

The BYOG toolkit is a set of scripts, templates, and procedures designed to help you configure a RedHat or SUSE Linux grid environment. For full details on the BYOG toolkit, refer to Appendix A, “Grid deployment toolkits” on page 243.

The steps in setting up the grid environment infrastructure using the BYOG toolkit are as follows:

1. Setting up NFS, NIS, SSH, and Ganglia on the front end node.
2. Preparing the PXE Boot server on the front end node.
3. PXE boot execution on a compute node.

Setting up NFS, NIS, SSH, and Ganglia on the front end node

NFS, NIS, SSH, and Ganglia software packages are required to support the grid environment. This step describes the installation and configuration of these software packages on the front end node.

Note: IBM Workload Scheduler LoadLeveler must also be installed. This is described in “Installing and configuring Workload Scheduler LoadLeveler” on page 102.

NFS

In the grid environment, compute nodes access IBM Information Server and other libraries such as Workload Scheduler LoadLeveler on the front node through NFS mounts. This requires that NFS be installed on the front end node, ensure it is started at boot, and exporting the shared file systems for mounting by the compute nodes. The following procedure details the steps to install NFS on the front end node:

1. The code in Example 2-31 checks to see if the NFS package is installed. Because no results were returned, it indicated that the NFS package was not installed.

Example 2-31 Check if the NFS package is installed

```
[root@orion ~]#rpm -qa|grep nfs
```

2. The code in Example 2-32 installs the NFS package.

Example 2-32 Install the NFS package

```
[root@orion ~]#rpm -ivh nfs-utils-lib* system-config-nfs* nfs-utils*
```

3. The code in Example 2-33 verifies that NFS was successfully installed.

Example 2-33 Verify that the NFS package is installed

```
[root@orion ~]# rpm -qa | grep nfs
nfs-utils-1.0.6-80.EL4
nfs-utils-lib-1.0.6-8
system-config-nfs-1.2.8-1
```

4. The code in Example 2-34 activates the NFS service and verifies that the NFS service is configured to start at system boot.

Example 2-34 Start the NFS service and verify

```
[root@orion ~]# chkconfig --level 345 nfs on

[root@orion ~]# chkconfig --list|grep nfs
nfs 0:off 1:off 2:off 3:on 4:on 5:on 6:off
```

Note: For more details on setting up NFS, refer to the following Web page:

<http://www.ibm.com/developerworks/systems/library/es-networkinstall/index.html>

The exports file (Example 2-35 on page 80) identifies the file systems that must be made available for mounting, and the various parameters that control the access clients have to these file systems. The /home, /ds_overview, and /opt/IBM file systems can be mounted read/write, while the /tftpboot can only be mounted read only.

Attention: In general, the software (/home, /ds_overview, and /opt/IBM file systems) is NFS-mounted to the compute nodes from either the front end node or from a NAS device. It is not always from the front end node. The only requirement for NFS on the front end is for the PXE boot process (which is normally defined as the server providing the software for the PXE boot images).

In our case, the front end node uses a SAN for the software and exposes it through NFS.

If we had a NAS configuration, we only require the /tftpboot directory in the exports file for NFS.

The **exportfs** command (Example 2-36 on page 80) exports all the file systems in the /etc/exports file to all the NFS clients.

Example 2-35 /etc/exports file contents

```
.....  
/home *(rw, sync)  
/ds_overview *(rw, sync)  
/opt/IBM *(rw, sync)  
/tftpboot *(ro)  
.....
```

Example 2-36 exportfs -a to update the NFS server

```
exportfs -a
```

NIS

Network Information Service (NIS) is a directory service that is used for maintenance and distribution of a central directory of user and group information, host names, e-mail aliases, and other text-based tables of information in a computer network.

Note: NIS was originally called Yellow Pages or YP.

The central directory eliminates the need to duplicate this information about the individual systems by maintaining a central database repository on a master server. One or more subordinate servers may be defined to ensure the continuity of NIS services in the event of the master server being down.

In a grid environment, the compute nodes and the standby node must share user account information and host names. This is achieved through NIS. For HA, NIS subordinate servers must be configured as follows:

1. The code in Example 2-37 checks to see if the NIS package `ypserv-2.13-18.i386.rpm` is installed. Because no results were returned, it indicated that the NIS package was not installed.

Example 2-37 Check if the NIS package is installed

```
rpm -qa | grep -i ypserv
```

2. The code in Example 2-38 installs the NIS package from the CDROM, while the code in Example 2-39 verifies that NIS was successfully installed.

Example 2-38 Install the NIS package

```
rpm -ivh /mnt/cdrom/Redhat/RPMS/ypserv-2.13-18.i386.rpm
Preparing... ##### [100%]
1:ypserv ##### [100%]
```

Example 2-39 Verify that the NIS package is installed

```
# rpm -qa | grep -i ypserv
ypserv-2.13-18
```

Example 2-40 shows the configuration of the NIS domain (IS.redbook.grid from Table 2-2 on page 63) by updating the network file in the /etc/sysconfig directory.

When an NIS client broadcasts its requests for information from an NIS server, it includes the name of the NIS domain of which it is part. This enables multiple NIS servers on a network to determine the server that must answer a particular request. An NIS domainname is the name for a group of related hosts. We strongly recommend that you do not choose the Internet domainname as the NIS domainname, because it can cause confusion when debugging network problems as well as result in security exposures.

Example 2-40 /etc/sysconfig/network file contents

```
.....
NETWORKING=yes
HOSTNAME=orionpvt.itsosj.sanjose.ibm.com
GATEWAY=9.43.85.1
NISDOMAIN=IS.redbook.grid
```

Example 2-41 shows the contents of the ypservers file in the /var/yp directory that lists the NIS servers in the namespace. It identifies the domain names on the private network for the front end node (ORIONPVT.itsosj.sanjose.ibm.com) and standby node (NILEPVT.itsosj.sanjose.ibm.com) as the NIS servers.

Note: The `ypinit -m` command updates this file.

Example 2-41 /var/yp/ypservers file contents

```
orionpvt.itsosj.sanjose.ibm.com
nilepvt.itsosj.sanjose.ibm.com
```

Example 2-42 shows the configuration of the Makefile in the /var/yp directory that tells NIS to push the map files to subordinate servers (NOPUSH=false) listed in the ypservers file (Example 2-41 on page 81). Our HA environment requirement had us define an NIS subordinate server on the standby node.

NIS maps are database files that are kept in the /var/yp directory. They are generated from configuration files in the /etc directory of the NIS master. The NIS database is composed of the following files:

- passwd.byname
- group.byname
- networks.byaddr
- hosts.byname
- protocols.bynumber
- services.byname
- mail.aliases
- ethers.byname

Example 2-42 /var/yp/Makefile file contents

```
....  
NOPUSH=false  
.....
```

3. The code in Example 2-43 sets the system's NIS/YP domain name to IS.redbook.grid using the **ypdomainname** command.

Note: It is possible for a single machine to be an NIS master server for more than one NIS domain, but only one domain can be current at any one time. This command sets the domain name to IS.redbook.grid for this machine.

Example 2-43 Set the ypdomainname

```
ypdomainname IS.redbook.grid
```

4. The code in Example 2-44 restarts the portmap service by issuing the **portmap restart** command.

Example 2-44 Restart the portmap service

```
portmap restart
```

About portmap: The portmap service is a dynamic port assignment daemon for RPC services such as NIS and NFS

5. The code in Example 2-45 activates and starts the following NIS services:
- ypserv: The ypserv daemon looks up information in local NIS maps
 - yppasswdd: The yppasswd daemon receives and executes requests from the yppasswd command
 - ypxfrd: The ypxfrd daemon is the transfer agent that handles map transfer requests from yppush on the master server to the subordinate server and clients
 - ypbind: The ypbind daemon enables client processes to bind, or connect, to an NIS server

Example 2-45 Configure and start the yp services

```
chkconfig --level 345 ypserv on
chkconfig --level 345 yppasswdd on
chkconfig --level 345 ypxfrd on
chkconfig --level 345 ypbind on
service ypserv start
service yppasswdd start
service ypxfrd start
service ypbind start
```

6. The code in Example 2-46 configures this NIS server as the master by issuing the **ypinit -m** command.

This converts /etc/passwd, /etc/shadow, hosts, and other files into NIS GNU dbm database format and generate a make file.

At the prompt, add both master (ORIONPVT.itsosj.sanjose.ibm.com) and subordinate server (NILEPVT.itsosj.sanjose.ibm.com).

Example 2-46 Configure NIS server as master

```
[root@orion post-install]# /usr/lib/yp/ypinit -m
At this point, we have to construct a list of the hosts which will
run NIS
servers. orion.itsosj.sanjose.ibm.com is in the list of NIS server
hosts. Please continue to add the names for the other hosts, one
per line.
When you are done with the
list, type a <control D>.
next host to add: orionpvt.itsosj.sanjose.ibm.com
next host to add: nilepvt.itsosj.sanjose.ibm.com
next host to add:
The current list of NIS servers looks like this:
orionpvt.itsosj.sanjose.ibm.com
nilepvt.itsosj.sanjose.ibm.com
```

```
Is this correct? [y/n: y] y
We need a few minutes to build the databases...
Building /var/yp/IS.redbook.grid/ypservers...
Running /var/yp/Makefile...
gmake[1]: Entering directory `/var/yp/IS.redbook.grid'
Updating passwd.byname...
Updating passwd.byuid...
Updating group.byname...
Updating group.bygid...
Updating hosts.byname...
Updating hosts.byaddr...
Updating rpc.byname...
Updating rpc.bynumber...
Updating services.byname...
Updating services.byservicename...
Updating netid.byname...
Updating protocols.bynumber...
Updating protocols.byname...
Updating mail.aliases...
gmake[1]: Leaving directory `/var/yp/IS.redbook.grid'

orionpvt.itsosj.sanjose.ibm.com has been set up as a NIS master
server.

Now you can run ypininit -s orionpvt.itsosj.sanjose.ibm.com on all
slave server.
```

Attention:

- ▶ On the NIS subordinate server (NILEPVT.itsosj.sanjose.ibm.com), you must issue the following command:

```
ypinit -s ORIONPVT.itsosj.sanjose.ibm.com
```

- ▶ On each of the NIS clients (LUXORPVT.itsosj.sanjose.ibm.com, TARUSPVT.itsosj.sanjose.ibm.com, and PHOENIXPVT.itsosj.sanjose.ibm.com), you must add the following entries to the /etc/yp.conf file:

```
domain IS.redbook.grid server 192.168.1.101
domain IS.redbook.grid server 192.168.1.76
```

The post installation process of the BYOG allows for additional configuration, which can update the /etc/yp.conf file during system build. A manual process is not required.

7. The code in Example 2-47 provides for rebuilding the NIS database and pushing configuration change notification to NIS clients when you make changes (such as adding users) to the NIS master server by issuing the **make -C /var/yp** command.

Example 2-47 make command to rebuild NIS database

```
make -C /var/yp
```

Because configuration changes may occur at any time, it is desirable to have the **make -C /var/yp** command issued at frequent intervals using the cron utility as shown in Example 2-48. In our case, we chose to run it once a day at 2 AM.

Example 2-48 Schedule cron utility

```
# crontab -e                                <<< edit the crontab file
# Run make -C /var/yp every day at 2 AM
0 2 * * * /usr/bin/make -C /var/yp
```

Note: For more details on setting up an NIS master server and NIS subordinate servers, refer to the following Web page:

<http://www.ibm.com/developerworks/aix/library/au-linuxtogether/index.html>

SSH

The IBM InfoSphere Enterprise Framework requires password-less login from the front end node to each defined compute node within the grid. An SSH key is required for each user that runs IBM Information Server and LoadLeveler. For IBM Information Server, the user is dsadm. For LoadLeveler, the user is loadl.

1. To generate the SSH key, change to the user that needs password-less login and generate an ssh key. Example 2-49 shows the generation for user dsadm.

Example 2-49 Generate ssh key for user dsadm

```
su - dsadm
ssh-keygen -t rsa
```

Note: You must repeat this procedure for each user that runs InfoSphere Applications.

2. Copy the generated key into the authorized keys file and remove execute permission as shown in Example 2-50.

Note: Because we used a shared directory for the /home directory, we do not have to copy it into the individual compute nodes' /home directory.

Example 2-50 Copy into the authorized key file and remove execute permission

```
cp ~/.ssh/id_rsa.pub ~/.ssh/authorized_keys
chmod 600 ~/.ssh/authorized_keys
```

3. Create a config file in the ~/.ssh/ directory with a single entry StrictHostKeyChecking no as shown in Example 2-51.

Example 2-51 ~/.ssh/config file contents

```
StrictHostKeyChecking no
```

Preparing the PXE Boot server on the front end node

This step describes the configuration of the Preboot eXecution Environment (PXE) boot server on the front end node that is used by the individual compute nodes to download, install, and configure the appropriate prerequisite software.

Attention: You do not have to choose PXE to install and configure your compute nodes if you happen to have a more efficient procedure to do so.

The PXE is an environment on which to boot computers using a network interface independently of available data storage devices such as a local disk drive or CDROMs. Booting through the network is often an effort to centralize management of disk storage, especially in cluster computing where nodes may or may not have local disks. The initial software to be loaded on a node is loaded from a server in the network, typically using the Trivial File Transfer Protocol (TFTP). The server from which to load the initial software is usually found by broadcasting or multicasting a Bootstrap Protocol or Dynamic Host Configuration Protocol (DHCP) request. Typically, this initial software is not a full image of the operating system to be loaded, but just enough for the operating system to start and then take control of the booting process, and continue booting over the network.

Note: For further details on PXE, refer to the following Web page:

<http://www.redhat.com/docs/manuals/enterprise/RHEL-4-Manual/sysadmin-guide/pt-install-info.html>

Setting up the front end node as a PXE Boot server involves the following steps:

1. Creating a tftpboot directory structure and verifying that it can be mounted as a file system.
2. Creating the RedHat installation tar image and copying it to the /tftpboot directory.
3. Installing BYOG tar file under the /tftpboot directory.
4. Verifying /tftpboot directory structure.
5. Modifying the postenv file.
6. Installing and configuring a DHCP service.
7. Installing and configuring a TFTP service.
8. Installing and configuring Ganglia.
9. Installing and configuring Workload Scheduler LoadLeveler.
10. Adding the compute nodes to Loadl_admin file.
11. Creating the kickstart file for each new compute node.

These steps are described in the following sections:

Creating a tftpboot directory structure and verifying that it can be mounted as a file system

Create the tftpboot directory structure and verify tftpboot directory structure and verify that it can be mounted as a file system because the PXE client accesses the contents of the /tftpboot directory in that manner.

1. The code in Example 2-52 creates the tftpboot structure required for setting up the PXE boot server environment on the front end node.

Example 2-52 Create tftpboot directory structure

```
mkdir /tftpboot
mkdir /tftpboot/linux-install
mkdir /tftpboot/linux-install/pxeboot
mkdir /tftpboot/linux-install/pxelinux.cfg
mkdir /tftpboot/config_template
mkdir /tftpboot/NICfiles
```

2. Verify that /tftpboot can be mounted as a file system by performing the following steps:
 - a. Establish a mount point /mnt/tmp as follows:


```
mkdir /mnt/tmp
```
 - b. Verify that /tftpboot can be mounted as a file system as shown in Example 2-53.

Example 2-53 Verify that /tftpboot can be mounted as a file system

```
[root@orion ~]# mount 192.168.1.101:/tftpboot /mnt/tmp
[root@orion ~]# df -k
```

Filesystem	1K-blocks	Used	Available	Use%	Mounted on
/dev/mapper/VolGroup00-LogVol00	574536368	49731048	495620504	10%	/
/dev/sda1	101086	15340	80527	17%	/boot
none	4154816	0	4154816	0%	/dev/shm
192.168.1.101:/tftpboot	574536384	49731040	495620512	10%	/mnt/tmp

- c. After successful validation, unmount the file system as follows:

```
umount /mnt/tmp
```

Creating the RedHat installation tar image and copying it to the /tftpboot directory

Create the Redhat4.tar.gz file in the /tftpboot directory, list it to verify its contents, and extract it to the /tftpboot directory.

Note: RedHat Linux Enterprise 4 is not provided as part of the BYOG toolkit. You must license it separately.

The Redhat4.tar.gz file is created from an install image on a DVD or CD and copied into the tftpboot directory using the following steps:

1. Mount the RedHat Linux Enterprise 4 DVD on /mnt/RedHat4.
2. Issue the command `cd /mnt` as root.
3. Issue the command `tar -cvzf /tftpboot/RedHat4.tar.gz ./.`
4. List the properties through the command `ls -l /tftpboot/Redhat4.tar.gz`.

These steps, commands, and output are shown in Example 2-54 on page 89.

Example 2-54 Redhat4.tar.gz file creation from install image into the /tftpboot directory

```
1) Mount the RedHat 4 DVD on /mnt/RedHat4
2) Run the following commands:
[root@orion /]# cd /mnt
[root@orion cdrom]# tar -cvzf /tftpboot/RedHat4.tar.gz .
./
./RELEASE-NOTES-U1-it.html
./RELEASE-NOTES-U1-es.html
./RELEASE-NOTES-en
./RELEASE-NOTES-U5-de.html
./RELEASE-NOTES-U3-ta.html
./README-zh_CN.html
./RELEASE-NOTES-U3-en.html
./RELEASE-NOTES-ru.html
./README-ja.html
./RELEASE-NOTES-U5-en.html
./RELEASE-NOTES-U4-fr.html
./RELEASE-NOTES-U1-en
./RELEASE-NOTES-U4-pa.html
./RELEASE-NOTES-U4-ta.html
./isolinux/
./isolinux/options.msg
./isolinux/initrd.img
./isolinux/TRANS.TBL
./isolinux/splash.lss
./isolinux/boot.msg
./isolinux/snake.msg
./isolinux/isolinux.cfg
./isolinux/memtest
./isolinux/isolinux.bin
./isolinux/vmlinuz
...
...
./RedHat/RPMS/authd-1.4.3-2.i386.rpm
./RedHat/RPMS/vsftpd-2.0.1-5.EL4.5.i386.rpm
./RedHat/RPMS/audit-libs-devel-1.0.15-3.EL4.i386.rpm
./RELEASE-NOTES-bn.html
./RELEASE-NOTES-U4-es.html
./RELEASE-NOTES-U2-bn.html
./RELEASE-NOTES-gu.html
./README-it.html
./RELEASE-NOTES-U4-pt_BR.html
./RELEASE-NOTES-U2-pa.html
./RELEASE-NOTES-U5-en
./RELEASE-NOTES-U4-de.html
./RELEASE-NOTES-U1-pt_BR.html
./RPM-GPG-KEY
```

```

./README-es.html
./RELEASE-NOTES-U2-it.html
./RELEASE-NOTES-es.html
./RELEASE-NOTES-zh_CN.html
./README-ta.html
./RELEASE-NOTES-fr.html
./RELEASE-NOTES-U2-hi.html
./RELEASE-NOTES-U1-ta.html
./RELEASE-NOTES-U5-ru.html

```

3) File produced is

```

[root@orion cdrom]# ls -l /tftpboot/Redhat4.tar.gz
-rw-r--r-- 1 root root 2192709112 Mar 22 13:42 /tftpboot/Redhat4.tar.gz

```

We then unpack the RedHat Linux Enterprise 4 tar image to the /tftpboot directory that is then downloaded by the boot image (transferred by the TFTP service earlier) on the PXE client as shown in Example 2-55.

Example 2-55 Unpack RedHat Linux Enterprise 4 to /tftpboot directory

```

[root@orion tftpboot]# tar -tvzf Redhat4.tar.gz
drwx----- root/root      0 2008-03-12 11:58:44 Redhat4/
-rwx----- root/root      99 2007-04-21 05:31:38 Redhat4/.discinfo
-rwx----- root/root     248 2007-04-11 13:27:28 Redhat4/autorun
-rwx----- root/root    6987 2004-12-16 11:42:26 Redhat4/EULA
-rwx----- root/root   18416 2004-12-16 11:42:26 Redhat4/GPL
drwx----- root/root      0 2008-03-12 11:58:45 Redhat4/images/
-rwx----- root/root  5718016 2007-04-21 02:36:56
Redhat4/images/boot.iso
...
-rwx----- root/root     1910 2004-12-16 11:42:26 Redhat4/RPM-GPG-KEY
-rwx----- root/root    29641 2007-04-21 07:26:40 Redhat4/TRANS.TBL

```

```

[root@orion tftpboot]# tar -xvzf /tftpboot/Redhat4.tar.gz

```

```

Redhat4/
Redhat4/.discinfo
Redhat4/autorun
Redhat4/EULA
Redhat4/GPL
Redhat4/images/
Redhat4/images/boot.iso
Redhat4/images/diskboot.img
Redhat4/images/pxeboot/
Redhat4/images/pxeboot/initrd.img
Redhat4/images/pxeboot/README
Redhat4/images/pxeboot/TRANS.TBL
Redhat4/images/pxeboot/vmlinuz

```



```
Redhat4/images/README
Redhat4/images/TRANS.TBL
...
Redhat4/RedHat/RPMS/bind-9.2.4-24.EL4.i386.rpm
Redhat4/RedHat/RPMS/bind-chroot-9.2.4-24.EL4.i386.rpm
Redhat4/RedHat/RPMS/bind-devel-9.2.4-24.EL4.i386.rpm
Redhat4/RedHat/RPMS/bind-libs-9.2.4-24.EL4.i386.rpm
Redhat4/RedHat/RPMS/bind-utils-9.2.4-24.EL4.i386.rpm
...
Redhat4/RedHat/RPMS/gcc-3.4.6-8.i386.rpm
Redhat4/RedHat/RPMS/gcc-c++-3.4.6-8.i386.rpm
Redhat4/RedHat/RPMS/gcc-g77-3.4.6-8.i386.rpm
Redhat4/RedHat/RPMS/gcc-gnat-3.4.6-8.i386.rpm
Redhat4/RedHat/RPMS/gcc-java-3.4.6-8.i386.rpm
Redhat4/RedHat/RPMS/gcc-objc-3.4.6-8.i386.rpm
Redhat4/RedHat/RPMS/gcc4-4.1.1-53.EL4.i386.rpm
Redhat4/RedHat/RPMS/gcc4-c++-4.1.1-53.EL4.i386.rpm
Redhat4/RedHat/RPMS/gcc4-gfortran-4.1.1-53.EL4.i386.rpm
Redhat4/RedHat/RPMS/gcc4-java-4.1.1-53.EL4.i386.rpm
Redhat4/RedHat/RPMS/gconf-editor-2.8.0-2.i386.rpm
```

Installing BYOG tar file under the /tftpboot directory

Extract the BYOG tar file under the /tftpboot directory as shown in Example 2-56.

Example 2-56 Extract BYOG tar file under /tftpboot directory

```
# cd /tftpboot
# tar -xvf /tmp/BYOG.2.1.tar
BYOG/
BYOG/define_IP.sh
BYOG/create_AI.sh
BYOG/create_ks.sh
BYOG/post-install/
BYOG/post-install/NFS
BYOG/post-install/NIS
BYOG/post-install/ganglia-web-3.0.2-1.noarch.rpm
BYOG/post-install/LoadL_initd
BYOG/post-install/LoadL
BYOG/post-install/Master
BYOG/post-install/gmetad_SuSe
BYOG/post-install/post-env
BYOG/post-install/ganglia-gmetad-3.0.2-1.i386.rpm
BYOG/post-install/ganglia_gmond
BYOG/post-install/ganglia-gmond-3.0.2-1.i386.rpm
BYOG/post-install/gmond_SuSe
```

```
BYOG/post-install/rrdtool-1.0.50-2.1.el3.rf.i386.rpm
BYOG/PXE_cfg.sh
BYOG/templates/
BYOG/templates/AutoInstall.xml
BYOG/templates/nsswitch.conf_Redhat
BYOG/templates/nsswitch.conf_SuSe
BYOG/templates/dhcpd.conf
BYOG/templates/kickstart
```

Verifying /tftpboot directory structure

Verify the /tftpboot structure after the installation of the BYOG toolkit as shown in Example 2-57.

Example 2-57 Verify /tftpboot structure

```
[root@orion /]# find /tftpboot -type d | sort
/tftpboot
/tftpboot/BYOG
/tftpboot/BYOG/post-install
/tftpboot/BYOG/templates
/tftpboot/config_template
/tftpboot/linux-install
/tftpboot/linux-install/pxeboot
/tftpboot/linux-install/pxelinux.cfg
/tftpboot/NICfiles
/tftpboot/Redhat4
/tftpboot/Redhat4/images
/tftpboot/Redhat4/images/pxeboot
/tftpboot/Redhat4/images/xen
/tftpboot/Redhat4/isolinux
/tftpboot/Redhat4/RedHat
/tftpboot/Redhat4/RedHat/base
/tftpboot/Redhat4/RedHat/RPMS
```

Modifying the postenv file

Modify the postenv file in the /tftpboot/BYOG/post-install directory as shown in Example 2-58. The values for the keywords are supplied from the information gathered in Table 2-2 on page 63.

Example 2-58 /tftpboot/BYOG/post-install/postenv file contents

```
FRONTEND_NAME=kazan.itsosj.sanjose.ibm.com
FRONTEND_IP=192.168.1.77
NIS_DOMAIN=IS.redbook.grid
NIS_MASTER_IP=192.168.1.101
NIS_SLAVE_IP=192.168.1.76
PACKAGES="NIS NFS Loadl ganglia_gmond"
export FRONTEND_NAME
export FRONTEND_IP
export NIS_DOMAIN
export NIS_MASTER_IP
export NIS_SLAVE_IP
export PACKAGES
```

The postenv file is used by the Master script to set these environment variables as appropriate. This Master script is invoked in the generated kickstart configuration file (shown in Example 2-98 on page 114 for compute node 198.168.1.102) for the post-installation process. The Master script processes all the necessary scripts defined in the postenv file using the environment variable "PACKAGES=NIS NFS Loadl ganglia_gmond".

Note: The BYOG toolkit provides the scripts for NIS, NFS, LoadL, ganglia_gmond, and other products. To add other products to the list, you have to develop the scripts to do so, and add them to the environment variable PACKAGES list within the postenv file.

Installing and configuring a DHCP service

Install and configure a DHCP service for sending PXE clients the IP address of the TFTP server and the location of the boot image on the TFTP server.

1. The code in Example 2-59 checks to see if the DHCP package is installed. Because no results were returned, it indicated that the DHCP package is not installed.

Example 2-59 Check if the DHCP package is installed

```
rpm -qa | grep dhcp
```

2. The code in Example 2-60 installs the DHCP package, while Example 2-61 on page 94 verifies that package was successfully installed.

Example 2-60 Install the DHCP package

```
rpm -ivh /tftpboot/Redhat4/RedHat/RPMS/dhcp-3.0.1-59.EL4.i386.rpm
```

```
Preparing... ##### [100%]  
1:dhcp ##### [100%]
```

Example 2-61 Verify that the DHCP package is installed

```
[root@orion tftpboot]# rpm -qa | grep dhcp  
dhcp-3.0.1-59.EL4
```

3. The configuration of the DHCP service is accomplished using two configuration files, `dhcpd.conf` and `dhcpd`, as follows:
 - `dhcpd.conf` file in the `/etc` directory identifies which IP addresses to assign and which server is providing a PXE boot image.
 - i. In the `/tftpboot/BYOG/templates` directory we have a template file named `dhcpd.conf` that must be copied over to the `/etc` directory (as shown in Example 2-63) after backing up the existing file as shown in Example 2-62.

Example 2-62 Backup the existing dhcpd.conf file in the /etc directory

```
cp /etc/dhcpd.conf /etc/dhcpd.conf.20080322
```

Example 2-63 Copy dhcpd.conf from /tftpboot/BYOG/templates directory to /etc directory

```
cp /tftpboot/BYOG/templates/dhcpd.conf /etc/dhcpd.conf
```

- ii. Modify the range dynamic-bootp range, next-server IP address and the filename as follows:

The range statement gives the lowest and highest IP addresses in a range.

The dynamic-bootp flag specifies that the addresses in the specified range may be dynamically assigned to BOOTP clients as well as DHCP clients. Because our compute nodes had IP addresses 192.168.1.102, 192.168.1.103 and 192.168.1.134, we chose to adopt the range 192.168.1.102 to 192.168.1.103 for this run (as highlighted in Example 2-64 on page 95) and perform the configuration for 192.168.1.134 later.

The next-server is used to specify the host address of the server from which the initial boot file (specified in the filename statement) is to be loaded, in our case: 192.168.1.101.

The filename specifies the initial boot file: /linux-install/pxelinux.0

Example 2-64 dhcpd.conf file: Modified contents

```
ddns-update-style interim;
ignore client-updates;

allow booting;
allow bootp;

subnet 192.168.1.0 netmask 255.255.255.0 {

    # --- default gateway
        option routers                192.168.1.101;
        option subnet-mask            255.255.255.0;

        option nis-domain              "IS.redbook.grid";
        option domain-name            "itsosj.sanjose.ibm.com";
        option domain-name-servers    192.168.1.101;

        option time-offset             -18000; # Eastern
Standard Time

        range dynamic-bootp 192.168.1.102 192.168.1.103;
        default-lease-time 21600;
        max-lease-time 43200;

        next-server 192.168.1.101;
        filename "/linux-install/pxelinux.0";

    }
```

- dhcpd file in the /etc/sysconfig directory contains information about which Ethernet interface provides the DHCP service.

In our case, we chose the private network which corresponds to the eth1 adapter.

Example 2-65 dhcpd file: Modified contents

DHCPDARGS=eth1

4. After the two files have been customized, configure the DHCP service to be enabled on reboot, restarted, and verified to be running as shown in Example 2-66 on page 96.

Example 2-66 Configure DHCP service

```
[root@orion tftpboot]# chkconfig --level 345 dhcpd on
[root@orion tftpboot]# service dhcpd restart
[root@orion tftpboot]# service dhcpd status
dhcpd (pid 2348) is running...
```

Installing and configuring a TFTP service

Install and configure a TFTP service for transferring a boot image from a DHCP server. Trivial File Transfer Protocol (TFTP) is a simple file transfer protocol that can be implemented with a small memory footprint. This makes it ideal for transferring small files (such as boot images) between hosts on a network.

1. The code in Example 2-67 checks to see if the TFTP package is installed. Because no results were returned, it indicates that the DHCP package is not installed.

Example 2-67 Check if the TFTP package is installed

```
rpm -qa | grep tftp
```

2. The code in Example 2-68 installs the TFTP package, while the code in Example 2-69 verifies that package was successfully installed.

Example 2-68 Install the TFTP package

```
rpm -ivh /tftpboot/Redhat4/RedHat/RPMS/ dhcp-3.0.1-59.EL4.i386.rpm

Preparing... ##### [100%]
1:tftp-server ##### [100%]
```

Example 2-69 Verify that the TFTP package is installed

```
[root@orion tftpboot]# rpm -qa | grep tftp
tftp-server-0.39-2
```

3. Enable the TFTP service. Edit the tftp file so that the service can be enabled. Change the value of the keyword disable from yes to no in the /etc/xinet.d/ directory. See Example 2-70 on page 97.

The TFTP service is configured by default to look into the /tftpboot directory as a relative path for files to be used in the PXE boot process.

```
#default: off
# description: The tftp server serves files using the trivial file
# transfer protocol. The tftp protocol is often used to boot
# diskless
# workstations, download configuration files to network-aware
# printers,
# and to start the installation process for some operating
# systems.
service tftp
{
  disable = no
  socket_type = dgram
  protocol = udp
  wait = yes
  user = root
  server = /usr/sbin/in.tftpd
  server_args = -s /tftpboot
  per_source = 11
  cps = 100 2
  flags = IPv4
}
```

4. After the tftp file has been enabled, configure the TFTP service to be enabled on reboot, restarted, and verified to be running as shown in Example 2-71.

```
[root@orion tftpboot]# chkconfig --level 345 xinetd on
[root@orion tftpboot]# chkconfig --level 345 tftp on
[root@orion tftpboot]# service xinetd restart
[root@orion tftpboot]# service xinetd status
xinetd (pid 2286) is running...
```

Installing and configuring Ganglia

Ganglia is a scalable distributed monitoring system for high-performance computing systems such as clusters and Grids. Based on a hierarchical design targeted at federations of clusters, it leverages widely-used technologies such as XML for data representation, XDR for compact, portable data transport, and RRDtool for data storage and visualization. It uses carefully engineered data structures and algorithms to achieve low per-node overheads and high concurrency. The implementation is robust, has been ported to an extensive set of operating systems and processor architectures, and is currently in use on

thousands of clusters around the world. It has been used to link clusters across university campuses and around the world and can scale to handle clusters with 2000 nodes.

About Ganglia: Ganglia is an open-source project that grew out of the University of California, Berkeley Millennium Project, initially funded in large part by the National Partnership for Advanced Computational Infrastructure (NPACI) and National Science Foundation RI Award EIA-9802069.

NPACI is funded by the National Science Foundation and strives to advance science by creating a ubiquitous, continuous, and pervasive national computational infrastructure: the Grid. Current support comes from Planet Lab, an open platform for developing, deploying, and accessing planetary-scale services.

We use Ganglia to monitor the frontend and compute nodes in our grid environment. Ganglia packages are provided as part of the BYOG toolkit as seen in Example 2-56 on page 91.

Example 2-72 through Example 2-76 on page 101 describe the installation and configuration of Ganglia on the front end node.

1. The code in Example 2-72 checks to see if the following Ganglia packages are installed:

- ganglia-gmetad-3.0.2-1.i386.rpm
- ganglia-gmond-3.0.2-1.i386.rpm
- ganglia-web-3.0.2-1.noarch.rpm
- rrdtool-1.0.50-2.1.el3.rf.i386.rpm

Because no results were returned, it indicated that the Ganglia packages were not installed.

Example 2-72 Check if the Ganglia packages are installed

```
rpm -qa | egrep -i 'ganglia|rrd'
```

2. The code in Example 2-73 on page 99 installs the Ganglia packages, while the code in Example 2-74 on page 99 verifies that the four packages were successfully installed.

Example 2-73 Install the Ganglia packages

```
rpm -ivh /tftpboot/BYOG/post-install/ganglia-gmetad-3.0.2-1.i386.rpm
rpm -ivh /tftpboot/BYOG/post-install/ganglia-gmond-3.0.2-1.i386.rpm
rpm -ivh /tftpboot/BYOG/post-install/ganglia-web-3.0.2-1.noarch.rpm
rpm -ivh /tftpboot/BYOG/post-install/rrdtool-1.0.50-2.1.el3.rf.i386.rpm

# rpm -Uvh ganglia-gmond-3.0.2-1.i386.rpm
Preparing... ##### [100%]
 1:ganglia-gmond ##### [100%]
Starting GANGLIA gmond: [ OK ]

# rpm -Uvh ganglia-gmetad-3.0.2-1.i386.rpm
Preparing... ##### [100%]
 1:ganglia-gmetad ##### [100%]
Starting GANGLIA gmetad: [ OK ]

# rpm -Uvh ganglia-web-3.0.2-1.noarch.rpm
Preparing... ##### [100%]
 1:ganglia-web ##### [100%]

# rpm -Uvh rrdtool-1.0.50-2.1.el3.rf.i386.rpm
warning: rrdtool-1.0.50-2.1.el3.rf.i386.rpm: V3 DSA signature: NOKEY, key ID 6b8d79e6
Preparing... ##### [100%]
 1:rrdtool ##### [100%]
```

Example 2-74 Verify that the Ganglia packages were installed

```
# rpm -qa | egrep -i 'ganglia|rrd'
rrdtool-1.0.50-2.1.el3.rf
ganglia-gmetad-3.0.2-1
ganglia-web-3.0.2-1
ganglia-gmond-3.0.2-1
```

3. Set the SELINUX to permissive mode by issuing the **setenforce 0** command.

Note: To see the charts on Ganglia, disable SELINUX on each node, including the front end node, with the appropriate **setenforce** command.

To put SELinux in permissive mode, enter the following command:

```
setenforce 0 (or Permissive)
```

To put SELinux in enforcing mode, enter the following command:

```
setenforce 1 (or Enforcing)
```

If you prefer not to set SELinux to permissive mode, you may also set httpd or rrdtool policies that allow the ganglia php scripts to access the required rrd lib directory and anything else that it might have to access.

4. Change the cluster name in the /etc/ directory from “unspecified” to “Data Integration” as highlighted in Example 2-75.

Example 2-75 /etc/gmond.conf file contents

```
/* This configuration is as close to 2.5.x default behavior as possible
   The values closely match ./gmond/metric.h definitions in 2.5.x */
globals {
    setuid = yes
    user = nobody
    cleanup_threshold = 300 /*secs */
}

/* If a cluster attribute is specified, then all gmond hosts are
   wrapped inside
   * of a <CLUSTER> tag. If you do not specify a cluster tag, then all
   <HOSTS> will
   * NOT be wrapped inside of a <CLUSTER> tag. */
cluster {
    name = "Data Integration"
}

/* The host section describes attributes of the host, like the location
   */
host {
    location = "unspecified"
}
.....
```

The code in Example 2-76 shows the Ganglia services being activated and started. It is configured not to start at boot.

Example 2-76 Configure and start the Ganglia services

```
chkconfig --level 2345 gmond off
chkconfig --level 2345 ganglia off
chkconfig --level 345 httpd on
service gmond start
service gmetad start
service httpd start
```

5. Test that Ganglia has been successfully installed by pointing the browser at the following URL (as shown in Figure 2-8 on page 102):

`http://9.43.86.77/ganglia`

Because we had only configured the front end node was configured, that is the only node monitored here.

Note: You must have gmond running on the compute nodes in order to monitor them.

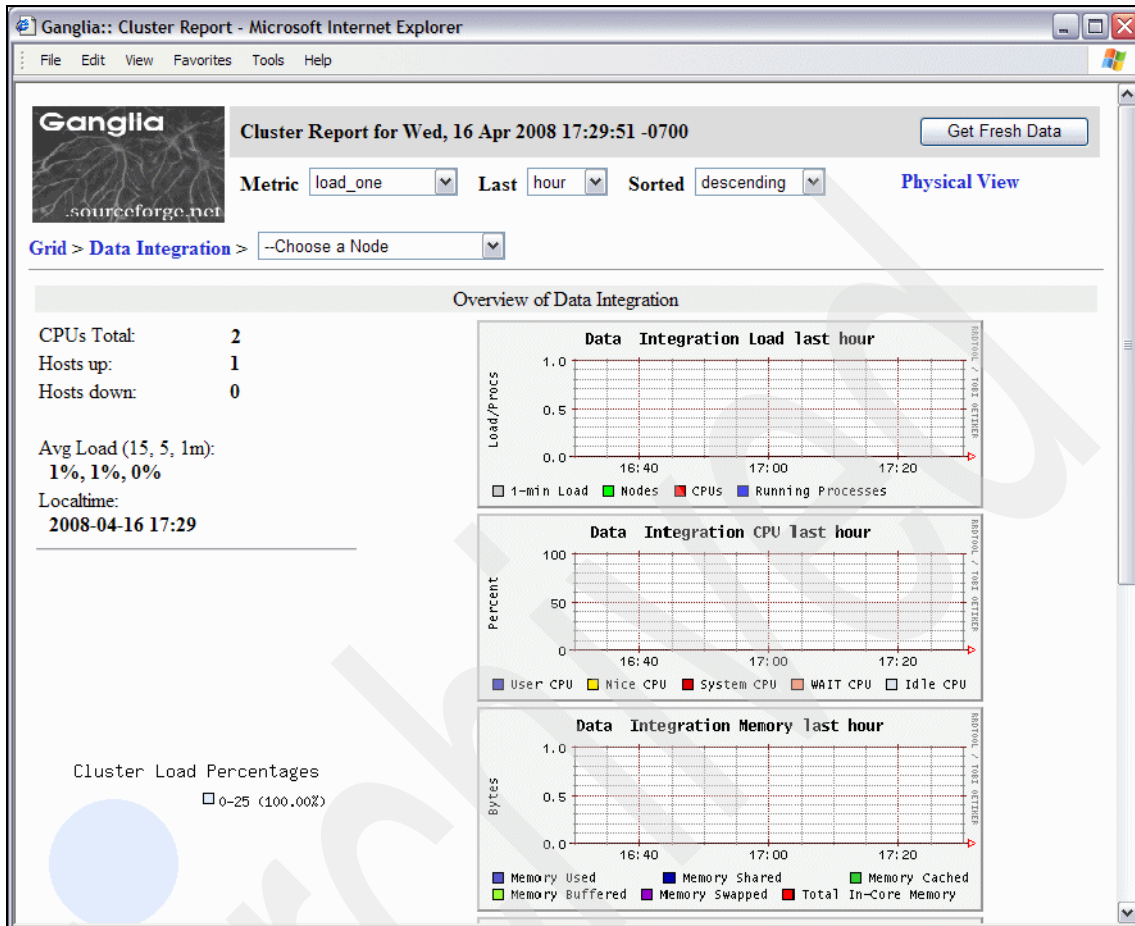


Figure 2-8 Ganglia

Installing and configuring Workload Scheduler LoadLeveler

Workload Scheduler LoadLeveler is a parallel job scheduling system that allows users to run more jobs in less time by matching each job's processing requirements and priority with the available resources, thereby maximizing resource use. LoadLeveler also provides a single point of control for effective workload management, offers detailed accounting of system use for tracking or chargeback, and supports HA configurations.

When jobs are submitted to LoadLeveler, they are not necessarily executed in the order of submission. LoadLeveler dispatches jobs based on their priority, resource requirements, and special instructions. For example, administrators can specify that long-running jobs run only on off-hours, that short-running jobs be scheduled around long-running jobs, or that certain users or groups get priority.

In addition, the resources themselves can be tightly controlled. Use of individual machines can be limited to specific times, users or job classes, or LoadLeveler can use machines only when the keyboard and mouse are inactive.

In a grid environment, the Grid Enablement Toolkit works with the resource manager (LoadLeveler) that manages the compute nodes.

Note: For details on installing Workload Scheduler LoadLeveler, refer to *IBM Tivoli Workload Scheduler LoadLeveler Version 3 Release 3.2: Installation Guide*, GI10-0763-02.

To install and configure LoadLeveler on the front end node, perform the following steps:

Note: LoadLeveler has to be installed under the loadl user.

1. Create the loadl group and user required for installing the LoadLeveler by using the code in Example 2-77.

Example 2-77 Create Workload Scheduler LoadLeveler group and user

```
groupadd loadl
useradd -d /home/loadl -m -g loadl loadl
```

2. Check to see if the Workload Scheduler LoadLeveler package LoadL-full-license-RH3-X86-3.3.2.0-0.i386.rpm is installed by using the code in Example 2-78. Because no results were returned, it indicated that the LoadL-full-license RPM package was not installed.

Example 2-78 Check if the Workload Scheduler LoadLeveler LoadL-full-license RPM package is installed

```
rpm -qa | grep -i loadl
```

3. Install the LoadLeveler LoadL-full-license RPM package in the /ftpboot/BYOG/post-install/ directory by using the code in Example 2-79 on page 104. Verify the successful installation of LoadLeveler LoadL-full-license RPM with the code in Example 2-80 on page 104.

Note: The Workload Scheduler LoadLeveler product is not provided with the BYOG toolkit. You must license this product and create a tar file named LoadLeveler.tar and copy it into the /tftpboot/BYOG/post-install directory. This procedure is beyond the scope of this Redbooks publication.

The Loadl script provided by the BYOG toolkit looks for a tar file named LoadLeveler.tar

Example 2-79 Install the Workload Scheduler LoadLeveler package

```
cd /tftpboot/BYOG/post-install/
tar -xvf /tftpboot/BYOG/post-install/LoadLeveler.tar
rpm -ivh /tftpboot/BYOG/post-install/LoadL-full-license-RH4-X86-3.3.2.0-0.i386.rpm

# rpm -ivh /tftpboot/BYOG/post-install/LoadL-full-license-RH4-X86-3.3.2.0-0.i386.rpm
Preparing...      ##### [100%]
 1:LoadL-full-license-RH4-##### [100%]
.....
```

Example 2-80 Verify that the Workload Scheduler LoadLeveler LoadL-full-license RPM package is installed

```
# rpm -qa | grep -i loadl
LoadL-full-RH4-X86-3.3.2.0-0
LoadL-full-license-RH4-X86-3.3.2.0-0
```

4. Create the directory for loadl local configuration with the code in Example 2-81.

Example 2-81 Create directory for loadl local configuration

```
mkdir /var/loadl
chown loadl:loadl /var/loadl
```

5. Install the Workload Scheduler LoadLeveler on the front end (ORION.itsosj.sanjose.ibm.com) identifying the initial software directory /tftpboot/BYOG/post-install with the code in Example 2-82.

Example 2-82 Install Workload Scheduler LoadLeveler

```
/opt/ibm11/LoadL/sbin/install_11 -y -d /tftpboot/BYOG/post-install
```

6. Initialize the LoadLeveler on the front end node (orion.itsosj.sanjose.ibm.com¹) under user loadl by running the linit script with the code in Example 2-83 on page 105.

¹ This has to be the physical host name of the front end node

Example 2-83 Initialize Workload Scheduler LoadLeveler on front end node

```
su - loadl
/opt/ibmll/LoadL/full/bin/llinit -local /var/loadl -release
/opt/ibmll/LoadL/full -cm
orion.itsosj.sanjose.ibm.com
```

Note: llinit initializes a new machine as a member of the LoadLeveler cluster by performing the following steps:

1. Creates the following LoadLeveler subdirectories with the given permissions:
 - spool subdirectory, with permissions set to 700.
 - execute subdirectory, with permissions set to 1777.
 - log subdirectory, with permissions set to 775.
 2. Copies the LoadL_config and LoadL_admin files from the release directory samples subdirectory into the loadl home directory.
 3. Copies the LoadL_config.local file from the release directory samples subdirectory into the local directory.
 4. Creates symbolic links from the loadl home directory to the spool, execute, and log subdirectories and the LoadL_config.local file in the local directory (if home and local directories are not identical).
 5. Creates symbolic links from the home directory to the bin, lib, man, samples, and include subdirectories in the release directory.
 6. Updates the LoadL_config with the release directory name.
 7. Updates the LoadL_admin with the central manager machine name.
7. After initialization, configure the LoadLeveler on the front end node as follows:
- a. Remove the LoadL_config file in the /home/loadl directory by removing loadladm from the LOADL_ADMIN entry as shown in Example 2-84.

Example 2-84 /home/loadl/LoadL_config file contents

```
#
# Machine Description
#
ARCH = i386
#
# Specify LoadLeveler Administrators here:
#
LOADL_ADMIN = loadl
.....
```

- b. Modify the Loadl_admin file in the /home/loadl directory by adding the host names of the compute nodes (phoenix, luxor, and tarus) after the <FrontendNode>: entry, where orion is the front end node as shown in Example 2-85. For each of the compute nodes, the central_manager is set to false and schedd_host is set to false.

Example 2-85 /home/loadl/LoadL_admin file contents

```
orion: type = machine
      central_manager = true
      schedd_host = true
phoenix: type = machine
      central_manager = false
      schedd_host = false
luxor: type = machine
      central_manager = false
      schedd_host = false
tarus: type = machine
      central_manager = false
      schedd_host = false
```

- c. Modify the LoadL_config.local file in the /home/loadl directory to prevent jobs from running on the front end node. Change the MAX_STARTERS value to 0 as shown in Example 2-86.

Example 2-86 /home/loadl/LoadL_config.local file contents

```
#
# file: LoadL_config.local
# Statements included here override on a PER MACHINE BASIS the
# statements in
# LoadL_config and may be used to tailor each machine individually.
# See samples in your RELEASEDIR/samples directory
#
START_DAEMONS= TRUE
MAX_STARTERS= 0

# Alternative ways of specifying the CLASS keyword
# the following is the old-style specification
#
#CLASS = { "small" "small" "small" "small" "small" "small" "small"
"small" "medium" "medium" "medium" "medium" "medium" "large" "large" }
# while the following is a newer, more concise specification
CLASS = small(8) medium(5) large(2) No_Class(1)
```

- d. Modify the `.bash_profile` (or `.profile`) in the `/home/loadl` directory to ensure that the `PATH` variable includes `/opt/ibm11/LoadL/full/bin`, then resource the profile as shown in Example 2-87.

Example 2-87 Resource /home/loadl/.bash_profile

```
# .bash_profile
# Get the aliases and functions
if [ -f ~/.bashrc ]; then
    . ~/.bashrc
fi
# User specific environment and startup programs
PATH=$PATH:$HOME/bin:/opt/ibm11/LoadL/full/bin
export PATH
unset USERNAME
```

8. After LoadLeveler has been configured, activate and start the LoadL service as root on the front end node, as shown in Example 2-88. It is configured not to start at boot. Verify that it is running by issuing the `llstatus` command under user `loadl`, as shown in Example 2-89.

Example 2-88 Configure and start Workload Scheduler LoadLeveler as root

```
cp /tftpboot/BYOG/post-install/LoadL_initd /etc/init.d/LoadL
chkconfig --add LoadL
chkconfig --level 345 LoadL off
service LoadL start
```

Example 2-89 Verify that Workload Scheduler LoadLeveler is running

```
[root@orion ~]# su - loadl
[loadl@orion ~]$ llstatus
```

Name	Schedd	InQ	Act	Startd	Run	LdAvg	Idle	Arch
OpSys								
luxor.itsosj.sanjose.ibm.	Avail		0	0 Down	0	1.10	24	i386
Linux2								
orion.itsosj.sanjose.ibm.	Avail		0	0 None	0	0.02	24	i386
Linux2								
phoenix.itsosj.sanjose.ib	Avail		0	0 Down	0	0.00	291	i386
Linux2								
tarus.itsosj.sanjose.ibm.	Avail		0	0 Down	0	0.00	24	i386
Linux2								
i386/Linux2		4 machines		0 jobs		0	running	
Total Machines		4 machines		0 jobs		0	running	

The Central Manager is defined on orion.itsosj.sanjose.ibm.com

The BACKFILL scheduler is in use

All machines on the machine_list are present.

Note: This step must be run for each of the machines (compute nodes) in the LoadLeveler cluster. This is performed as part of a post-install process of PXE boot as described in “PXE boot execution on a compute node” on page 116.

Adding the compute nodes to Loadl_admin file

All the compute nodes that are added to the grid in your current configuration (In our case, three compute nodes: phoenix, luxor, and tarus) must be manually added to the Loadl_admin file in the /home/load directory as shown in Example 2-90. The front end node is orion, which has the keyword central_manager = true in the Loadl_admin file.

Note: The Loadl_admin file may have nodes added that have not yet been installed and configured by the PXE boot execution procedure or any other procedure you may choose to set up the compute nodes. The PXE boot execution procedure is described in “PXE boot execution on a compute node” on page 116,.

Example 2-90 /home/loadl/LoadL_admin file

```
.....
orion:type = machine
      central_manager = true
      schedd_host = true
phoenix:type = machine
      central_manager = false
      schedd_host = false
luxor:type = machine
      central_manager = false
      schedd_host = false
tarus:type = machine
      central_manager = false
      schedd_host = false
.....
```

Creating the kickstart file for each new compute node

Red Hat provides system administrators an automated installation method for installing Red Hat Enterprise Linux on their machines. This is called the kickstart installation method, and requires the system administrator to create a text file containing responses to questions that are normally asked during a typical installation. Kickstart files can be kept on a central server system and read by individual computers during the installation. This method can support the use of a single kickstart file to install Red Hat Enterprise Linux on multiple machines.

In this step, we create separate kickstart files for each compute node on the front end node. These kickstart files are then referenced during PXE boot execution through a script within the BYOG toolkit and a template kickstart file.

We first modify a kickstart template file provided with the BYOG toolkit to support our environment. We then create PXE boot configuration files for each of the IP addresses associated with each compute node. The PXE boot configuration files are then input to the generation of a kickstart file for each compute node. To accomplish this, perform the following steps:

1. Modify the kickstart file provided with the BYOG toolkit that is used by PXE boot to install the Linux OS as follows:
 - a. Copy the original anaconda-ks.cfg file in the /root directory of the front end node as kickstart_template as shown in Example 2-91. If the /root/anaconda-ks.cfg file does not exist, you can copy it from the /tftpboot/BYOG/templates directory.

Example 2-91 Anaconda-ks.cfg file in the /root directory as kickstart_template

```
[root@luxor /tftpboot]# cd /tftpboot/config_template
[root@luxor config_template]# cp /root/anaconda-ks.cfg
kickstart_template
```

- b. Modify the kickstart_template file as highlighted in Example 2-92 on page 110:
 - i. Comment out the cdrom entry.
 - ii. Replace xconfig (commented out) with skipx.
 - iii. Ensure the presence of only one network entry in this file.

We provided the following keyword information for our environment:

- --host name
- --ip
- --netmask
- --gateway
- --nameserver

This is ignored, however, because we had more than one NIC connection, and this information is picked up from a network.# file (“#” being the last number of the IP address for the compute node) you create rather than from the network entry in this kickstart_template file. This is covered later.

Note: Because our environment has two NIC connections, we created this information through a network.# for each NIC connection.

- iv. Change the keyword selinux to permissive to support Ganglia.
- v. Modify the keyword --nisdomain to the string IS.redbook.grid
- vi. Ensure the time zone value is America/Los_Angeles (PST).
- vii. Add the comment Zero Master Boot Record and the entry zerombr yes.
- viii. Uncomment partition information specific to our server hardware (clearpart, and part entries)
- ix. Add the nfs entry with the keywords --server 192.168.1.101 and --dir /ftpboot/Redhat4 to find the Linux installation files.
- x. Add everything under and including %post --interpreter /bin/sh

Note: The reboot keyword was added to make sure components are started and there is a clean boot after the initial creation.

Example 2-92 /root/kickstart_template file: Modified contents

```
# Kickstart file automatically generated by anaconda.

install
# cdrom
lang en_US.UTF-8
langsupport --default=en_US.UTF-8 en_US.UTF-8
keyboard us
#xconfig --card "S3 Savage4" --videoram 8192 --hsync 31.5-37.9 --vsync 50-70
--resolution 800x600 --depth 16 --startxonboot --defaultdesktop gnome
skipx
network --device eth0 --bootproto static --hostname compute# --ip 9.43.86.77 --netmask
255.255.252.0 --gateway 9.43.85.1 --nameserver 9.43.85.11
rootpw --iscrypted $1$0fBvET4j$A/eH5IfbdkMMrGWyOoF6D/
firewall --disabled
selinux --permissive
authconfig --enablesshadow --enablemd5 --nisdomain IS.redbook.grid --enablenis
timezone America/Los_Angeles

### Zero Master Boot Record
zerombr yes

# The following is the partition information you requested
# Note that any partitions you deleted are not expressed
```

```

# here so unless you clear all partitions first, this is
# not guaranteed to work
cleartpart --all --drives=sda,sdb
part /boot --fstype "ext3" --size=100 --ondisk=sda
part pv.3 --size=0 --grow --ondisk=sda
part pv.4 --size=0 --grow --ondisk=sdb
volgroup VolGroup00 --pesize=32768 pv.3 pv.4
logvol / --fstype ext3 --name=LogVol00 --vgname=VolGroup00 --size=1024 --grow
logvol swap --fstype swap --name=LogVol01 --vgname=VolGroup00 --size=1000 --grow
--maxsize=2000

### Start bootloader here
bootloader --location=mbr --append="rhgb quiet"
###
# Added the nfs --server to find the Linux installation files
#
nfs --server 192.168.1.101 --dir /tftpboot/Redhat4
###
# Reboot after loading
reboot
%packages
@ smb-server
@ base-x
@ web-server
@ development-tools
@ printing
@ text-internet
@ admin-tools
@ server-cfg
@ network-server
@ compat-arch-support
@ system-tools
@ gnome-desktop
@ authoring-and-publishing
@ ftp-server
kernel
lvm2
grub
kernel-smp
vnc
tsclient
e2fsprogs

%post --interpreter /bin/sh
mkdir -p /mnt/tmp
mount 192.168.1.101:/tftpboot /mnt/tmp
cd /mnt/tmp/BYOG/post-install
. Master
cd /
umount /mnt/tmp
rmdir /mnt/tmp
### Adding reboot here
reboot

```

2. Create PXE boot configuration files based on IP addresses.

Create the PXE boot configuration files based on IP addresses, where the content of the file is used to determine which kickstart (Redhat) or AutoInstall (SUSE) file is used for the actual build. This step is required to tie all the pieces together. It maps compute node IP addresses to their PXE boot configuration files.

Note: Before creating the PXE boot configuration files, validate that only one row for a range of IP addresses exists in the `/etc/dhcpd.conf` file. See Example 2-93. This is critical, as the `PXE_cfg.sh` script that generates the PXE boot configuration files from the `dhcpd.conf` file is dependant on this.

Example 2-93 Verify only one row for a range of IP addresses exists in `/etc/dhcpd.conf`

```
[root@orion post-install]# grep range /etc/dhcpd.conf
range dynamic-bootp 192.168.1.102 192.168.1.103;
```

The PXE boot configuration files are generated by executing the `PXE_cfg.sh` script as shown in Example 2-94 on page 113. Because our range of IP addresses was 192.168.1.102 to 192.168.1.103, two configuration file C0A80166 (hex value of IP address 192.168.1.102) and C0A80167 (hex value of IP address 192.168.1.103) were generated.

Note: The third compute node corresponding to 192.168.1.134 is not shown here because it was configured in a separate step.

We might have used an IP address of 192.168.1.104 for the third compute node and had the range `dynamic-bootp 192.168.1.102 192.168.1.104`. This allows us to configure all three nodes in the same cycle. There is no requirement for the last number to be the same, but we chose to do so for the private and public addresses in our environment.

The configuration files are checked for validity by ensuring that the front-end IP address (192.168.1.101) and path (`/tftpboot/config_template`) is correct as shown in Example 2-95 on page 113 for 192.168.1.103. This file also references the name of the kickstart file (`kickstart103.cfg`) to be generated. When multiple NICs are involved, you are prompted to choose the device while running the `PXE_cfg.sh` script. Because we chose the private network for PXE boot, we identified `eth1`.

Example 2-94 Create PXE boot config files using PXE_cfg.sh toolkit script

```
[root@orion pxelinux.cfg]# sh /tftpboot/BYOG/PXE_cfg.sh
Ethernet device to PXE boot from (ie eth0 or eth1) default eth0: eth1
Creating COA80167 for 192.168.1.103
Creating COA80166 for 192.168.1.102
```

```
[root@orion pxelinux.cfg]# ls -l
total 8
-rw-r--r-- 1 root root 188 Apr  4 11:47 COA80166
-rw-r--r-- 1 root root 188 Apr  4 11:47 COA80167
```

Example 2-95 COA80167 config file contents

```
[root@orion pxelinux.cfg]# cat COA80167
default pxeboot

label pxeboot
    kernel pxeboot/vmlinuz
    append initrd=pxeboot/initrd.img console=tty0 ramdisk=10000
network ks=nfs:192.168.1.101:/tftpboot/config_template/kickstart103.cfg
ksdevice=eth1
```

3. Generate the kickstart file for each compute node as follows:

- a. Because our compute nodes have multiple NIC connections, we first must create a separate file called `network.#`, where “#” is the last number of the IP address for the compute node. This file provides network information about this compute node and overrides the network entry in the `kickstart_template` shown in Example 2-92 on page 110. The contents of such a file (`network.103`) is shown in Example 2-96 on page 114 for the compute node with IP address ending in 103. It shows the network information about the `eth1` and `eth0` Ethernet interfaces.
- b. With the modifications to the `kickstart_template` file, generation of the PXE boot configuration files, and the creation of the `network.#` files, we can create the actual kickstart configuration files for each compute node by executing the `create_ks.sh` script.

Example 2-97 on page 114 shows the execution of the `create_ks.sh` script that generates the `kickstart102.cfg` and `kickstart103.cfg` kickstart files (contents shown in Example 2-98 on page 114).

Note: When the `network.#` file is present, its contents are inserted verbatim into the kickstart configuration file created by the `create_ks.sh` script. See Example 2-98 on page 114.

In the kickstart103.cfg file shown in Example 2-98, the Master script is called at the end of the installation on the compute node. It runs on the new compute node executing any command in it. This is the mechanism by which the compute node is configured and customized for running on the Grid.

Example 2-96 network.103 file

```
network --device eth1 --bootproto static --ip 192.168.1.103 --netmask
255.255.252.0 --gateway 9.43.85.1 --nameserver 9.43.85.11 --hostname
taruspvt.itsosj.sanjose.ibm.com
network --device eth0 --bootproto static --ip 9.43.86.103 --netmask
255.255.252.0 --gateway 9.43.85.1 --nameserver 9.43.85.11 --hostname
tarus.itsosj.sanjose.ibm.com
```

Example 2-97 Generate kickstart files using the create_ks.sh script

```
[root@orion config_template]# sh /tftpboot/BYOG/create_ks.sh
Use -L to increment compute node names from low to high for network
default is to assign from 192.168.1.103 = 1 to 192.168.1.102 = <last>
-L assigns from 192.168.1.102 = 1 to 192.168.1.103 = <last> (low to
high)
Creating kickstart file for 192.168.1.103
Creating kickstart file for 192.168.1.102
[root@orion config_template]#
```

Example 2-98 kickstart103.cfg configuration file

```
# Kickstart file automatically generated by anaconda.

install
# cdrom
lang en_US.UTF-8
langsupport --default=en_US.UTF-8 en_US.UTF-8
keyboard us
#xconfig --card "S3 Savage4" --videoram 8192 --hsync 31.5-37.9 --vsync
50-70 --resolution 800x600 --depth 16 --startxonboot --defaultdesktop
gnome
skipx
network --device eth1 --bootproto static --ip 192.168.1.103 --netmask
255.255.252.0 --gateway 9.43.85.1 --nameserver 9.43.85.11 --hostname
taruspvt.itsosj.sanjose.ibm.com
network --device eth0 --bootproto static --ip 9.43.86.103 --netmask
255.255.252.0 --gateway 9.43.85.1 --nameserver 9.43.85.11 --hostname
tarus.itsosj.sanjose.ibm.com
rootpw --iscrypted $1$0fBvET4j$A/eH5IfbdkMMrGWyOoF6D/
firewall --disabled
```



```

selinux --permissive
authconfig --enableshadow --enablemd5 --nisdomain IS.redbook.grid
--enablenis
timezone America/Los_Angeles

### Zero Master Boot Reocrd
zerombr yes

# The following is the partition information you requested
# Note that any partitions you deleted are not expressed
# here so unless you clear all partitions first, this is
# not guaranteed to work
clearpart --all --drives=sda,sdb
part /boot --fstype "ext3" --size=100 --ondisk=sda
part pv.3 --size=0 --grow --ondisk=sda
part pv.4 --size=0 --grow --ondisk=sdb
volgroup VolGroup00 --pesize=32768 pv.3 pv.4
logvol / --fstype ext3 --name=LogVol00 --vgname=VolGroup00 --size=1024
--grow
logvol swap --fstype swap --name=LogVol01 --vgname=VolGroup00
--size=1000 --grow --maxsize=2000

### Start bootloader here
bootloader --location=mbr --append="rhgb quiet"
###
# Added the nfs --server to find the Linux installation files
#
nfs --server 192.168.1.101 --dir /tftpboot/Redhat4
###
# Reboot after loading
reboot
%packages
@ smb-server
@ base-x
@ web-server
@ development-tools
@ printing
@ text-internet
@ admin-tools
@ server-cfg
@ network-server
@ compat-arch-support
@ system-tools
@ gnome-desktop
@ authoring-and-publishing

```

```
@ ftp-server
kernel
lvm2
grub
kernel-smp
vnc
tsclient
e2fsprogs

%post --interpreter /bin/sh
mkdir -p /mnt/tmp
mount 192.168.1.101:/tftpboot /mnt/tmp
cd /mnt/tmp/BYOG/post-install
. Master
cd /
umount /mnt/tmp
rmdir /mnt/tmp
### Adding reboot here
reboot
```

PXE boot execution on a compute node

This step describes the PXE boot execution on a PXE client¹ that downloads, installs, and configures the appropriate prerequisite software.

All new systems must have the PXE boot method available in the BIOS. During a typical system startup, a boot program is loaded and executed prior to the actual OS being loaded. Normally this boot image comes from the first few blocks of the local disk drive. If no boot image is found on the local disk drive, however, and the PXE boot option is enabled within the BIOS, a network boot is performed. You can also change the boot order to perform a one time boot from the network by pressing F12 on the keyboard. PXE booting can also be used to reload nodes after applying patches on the conductor node.

The PXE client/server protocol is a combination of DHCP and TFTP, where DHCP is used to locate the boot server and TFTP is used to download the initial boot image, any other files that then bootstraps the operating system, and any other software.

When you power up a compute node, press F12, and choose a network boot, the BIOS sends out a DHCP request. The DHCP server's responsibility is to provide an IP address and the location of the boot program. This boot program is then responsible for loading the actual OS kernel into memory.

¹ Refers to the role that the compute node takes in the PXE boot process.

Note: For full details on PXE execution protocol, refer to the following Web page:

<http://www.redhat.com/docs/manuals/enterprise/RHEL-4-Manual/sysadmin-guide/pt-install-info.html>

In this section, we describe the PXE boot execution of the compute node designated as tarus.itsosj.sanjose.ibm.com with the IP Address 9.43.86.103.

1. Power up the compute node and press F12 to select the boot device as shown in Figure 2-9.

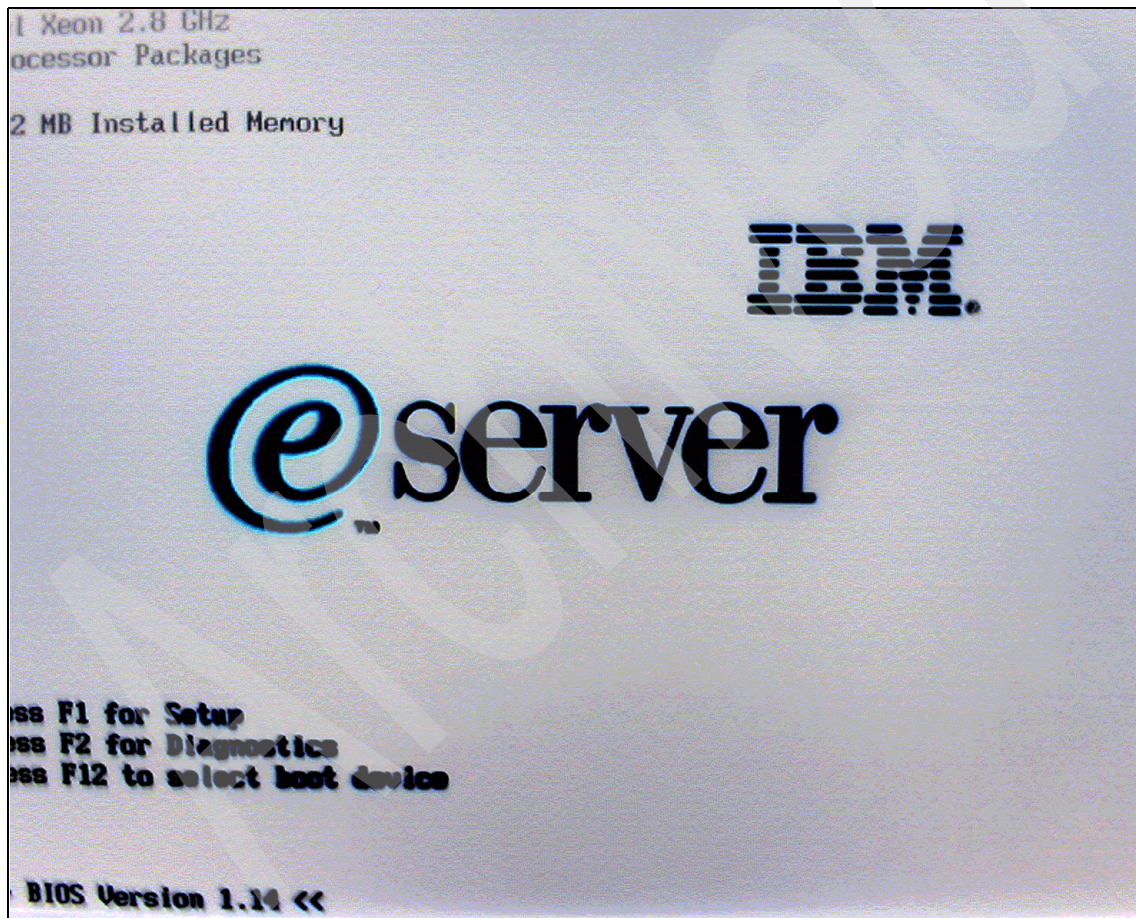


Figure 2-9 PXE boot execution, part 1 of 6

2. Select the Network boot option as shown in Figure 2-10.



Figure 2-10 PXE boot execution, part 2 of 6

Figure 2-11 on page 119 shows the compute node looking for a DHCP server.

This compute node has two NICs that we had discovered earlier and recorded in Table 2-2 on page 63:

- eth0 with MAC ADDR 00:14:5E:A4:E4:C6
- eth1 with MAC ADDR 00:14:5E:A4:E4:C7

The messages in Figure 2-11 on page 119 show that the BIOS did not find a DHCP server on MAC ADDR 00:14:5E:A4:E4:C6.

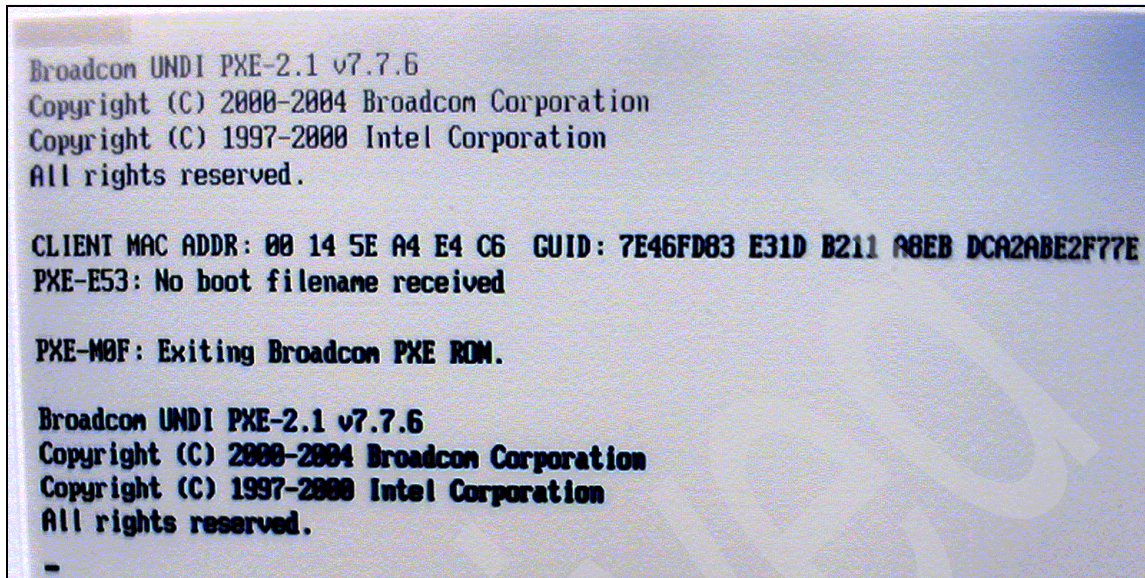


Figure 2-11 PXE boot execution, part 3 of 6

The BIOS then looks for a DHCP server on MAC ADDR 00:14:5E:A4:E4:C7. It finds the DHCP server as shown in Figure 2-12 on page 120. It then sets the IP address of the compute node 192.168.1.103.

```
Broadcom UNDI PXE-2.1 v7.7.6
Copyright (C) 2000-2004 Broadcom Corporation
Copyright (C) 1997-2000 Intel Corporation
All rights reserved.

CLIENT MAC ADDR: 00 14 5E A4 E4 C7  GUID: 7E46FD83 E31D B211 A8EB DCA2ABE2F7
CLIENT IP: 192.168.1.103  MASK: 255.255.255.0  DHCP IP: 192.168.1.101
GATEWAY IP: 192.168.1.101

PXELINUX 2.11 2004-08-16  Copyright (C) 1994-2004 H. Peter Anvin
UNDI data segment at:  00091D90
UNDI data segment size: 4D90
UNDI code segment at:  00096B20
UNDI code segment size: 4E7C
PXE entry point found (we hope) at 96B2:0006
My IP address seems to be 00000167 192.168.1.103
ip=192.168.1.103:192.168.1.101:192.168.1.101:255.255.255.0
TFTP prefix: /linux-install/
Trying to load: pxelinux.cfg/01-00-14-5e-a4-e4-c7
Trying to load: pxelinux.cfg/00000167
Loading pxeboot/vmlinuz.....
Loading pxeboot/initrd.img.....
*****
Ready.
```

Figure 2-12 PXE boot execution, part 4 of 6

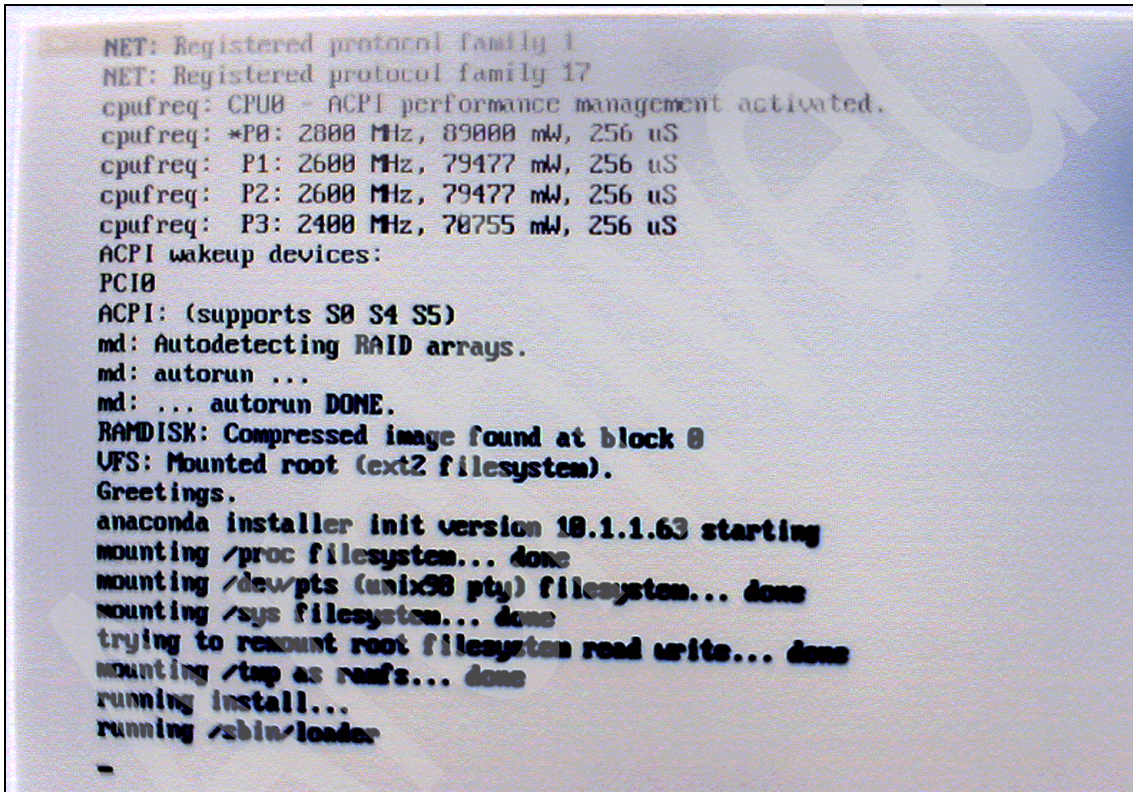
Important: The DHCP server refers to the dhcpd.conf file (Example 2-64 on page 95) and counts backwards from the range dynamic-bootp 192.168.1.102 192.168.1.103 entry. Because this was the first PXE client, the IP address selected was 192.168.1.103 and that is the one assigned. When the next PXE client is powered up, the next lower IP address is assigned to that compute node. You must power up the machines in the appropriate sequence if you want to assign specific IP addresses to specific machines.

It is possible to assign specific IP addresses to a particular MAC Address of an Ethernet adapter. A discussion of this is beyond the scope of this Redbooks publication.

The compute node is now booting from the TFTP prefix of /linux-install and looking for the files pxelinux.cfg/01-00-14-5e-a4-e4-c7 and pxelinux.cfg/C0A80167.

It finds the pxelinux.cfg/C0A80167 file (this is the PXE configuration file corresponding to the 192.168.1.103 IP address in hex that we had created earlier as shown in Example 2-95 on page 113) and starts loading the images identified in this file such as pxeboot/vmlinuz and pxeboot/initrd.img. It then runs the kickstart103.cfg identified in the pxelinux.cfg/C0A80167 file.

Figure 2-13 and Figure 2-14 on page 122 show the progress of the install.



```
NET: Registered protocol family 1
NET: Registered protocol family 17
cpufreq: CPU0 - ACPI performance management activated.
cpufreq: *P0: 2800 MHz, 89000 mW, 256 uS
cpufreq: P1: 2600 MHz, 79477 mW, 256 uS
cpufreq: P2: 2600 MHz, 79477 mW, 256 uS
cpufreq: P3: 2400 MHz, 70755 mW, 256 uS
ACPI wakeup devices:
PCI0
ACPI: (supports S0 S4 S5)
md: Autodetecting RAID arrays.
md: autorun ...
md: ... autorun DONE.
RAMDISK: Compressed image found at block 0
UFS: Mounted root (ext2 filesystem).
Greetings.
anaconda installer init version 10.1.1.63 starting
mounting /proc filesystem... done
mounting /dev/pts (unix98 pts) filesystem... done
mounting /sys filesystem... done
trying to remount root filesystem read write... done
mounting /tmp as ramfs... done
running install...
running /sbin/loader
-
```

Figure 2-13 PXE boot execution, part 5 of 6

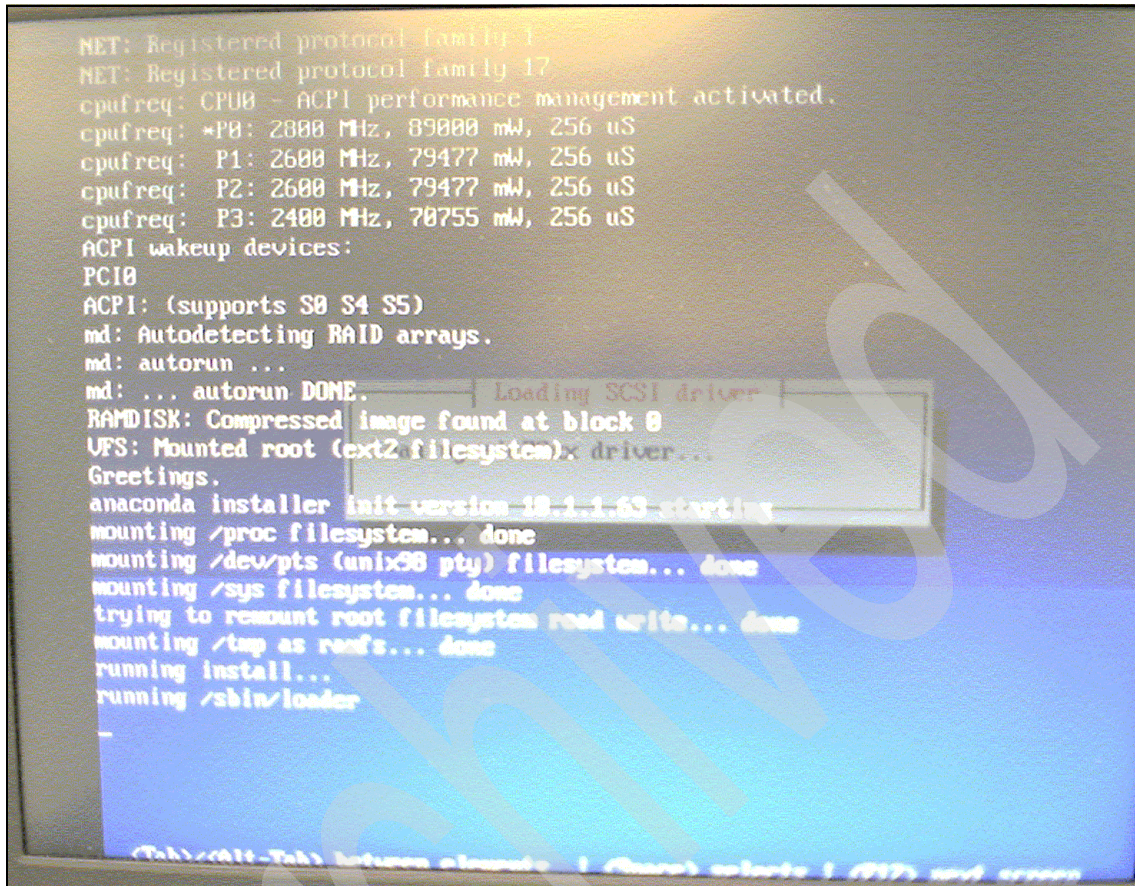


Figure 2-14 PXE boot execution, part 6 of 6

At the end of the PXE boot process, you must review some of the logs (Table 2-3) and files as follows:

Table 2-3 Summary of logs available for review

Brief description of the log	Log file location
Front end node and compute node system log messages	/var/log/messages
Compute node post install logs produced by the Master script	
Post install	/root/post-install
Workload Scheduler LoadLeveler	/root/LoadL.log

Brief description of the log	Log file location
NIS	/root/NIS.log
NFS	/root/NFS.log
Ganglia	/root/ganglia_gmond.log

- Example 2-99 and Example 2-100 show some of the messages (/var/log/messages file) that appear on the front end node where the DHCP server is installed.
 - Example 2-99 shows that the DHCP server has received DHCPDISCOVER request from 00:14:5e:a4:e4:c7 through its eth1 adapter. It responded by offering the IP address 192.168.1.103 to the compute node.

Example 2-99 DHCP and TFTP server messages in /var/log/messages, part 1 of 2

```

Apr  9 14:41:13 orion dhcpd: DHCPDISCOVER from 00:14:5e:a4:e4:c7 via eth1
Apr  9 14:41:14 orion dhcpd: DHCPOFFER on 192.168.1.103 to 00:14:5e:a4:e4:c7
via eth1
Apr  9 14:41:14 orion dhcpd: DHCPDISCOVER from 00:14:5e:a4:e4:c7 via eth1
Apr  9 14:41:14 orion dhcpd: DHCPOFFER on 192.168.1.103 to 00:14:5e:a4:e4:c7
via eth1
Apr  9 14:41:14 orion dhcpd: DHCPREQUEST for 192.168.1.103 (192.168.1.101) from
00:14:5e:a4:e4:c7 via eth1
Apr  9 14:41:14 orion dhcpd: DHCPACK on 192.168.1.103 to 00:14:5e:a4:e4:c7 via
eth1

```

- Example 2-100 shows the next request to the DHCP server from the new compute node comes four minutes later for tftpboot mount requests in order to load and install the Linux Redhat image.

Example 2-100 DHCP and TFTP server messages in /var/log/messages, part 2 of 2

```

Apr  4 18:54:09 orion mountd[6458]: authenticated mount request from
192.168.1.103:629 for /tftpboot/config_template (/tftpboot)
Apr  4 18:54:11 orion mountd[6458]: authenticated mount request from
192.168.1.103:633 for /tftpboot/Redhat4 (/tftpboot)
Apr  4 19:06:25 orion mountd[6458]: authenticated mount request from
192.168.1.103:739 for /tftpboot (/tftpboot)

```

- Example 2-101 shows the contents of the post install log on the compute node such as the ifconfig command output and the progress of the execution of the NIS, NFS, Loadl, and ganglia_gmond scripts.

Example 2-101 Post install log (/root/post-install) on the compute node

Wed Apr 9 17:40:02 PDT 2008

Running ifconfig

```
eth1      Link encap:Ethernet  HWaddr 00:14:5E:A4:E4:C7
          inet addr:192.168.1.103  Bcast:0.0.0.0  Mask:255.255.252.0
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:575419 errors:0 dropped:0 overruns:0 frame:0
          TX packets:29510 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:841382100 (802.4 MiB)  TX bytes:5123748 (4.8 MiB)
          Interrupt:169
```

```
lo        Link encap:Local Loopback
          inet addr:127.0.0.1  Mask:255.0.0.0
          UP LOOPBACK RUNNING  MTU:16436  Metric:1
          RX packets:54 errors:0 dropped:0 overruns:0 frame:0
          TX packets:54 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:0
          RX bytes:4452 (4.3 KiB)  TX bytes:4452 (4.3 KiB)
```

ifup eth0

Running ifconfig

```
eth0      Link encap:Ethernet  HWaddr 00:14:5E:A4:E4:C6
          inet addr:9.43.86.103  Bcast:9.43.87.255  Mask:255.255.252.0
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:3 errors:0 dropped:0 overruns:0 frame:0
          TX packets:1 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:242 (242.0 b)  TX bytes:64 (64.0 b)
          Interrupt:169
```

```
eth1      Link encap:Ethernet  HWaddr 00:14:5E:A4:E4:C7
          inet addr:192.168.1.103  Bcast:0.0.0.0  Mask:255.255.252.0
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:575434 errors:0 dropped:0 overruns:0 frame:0
          TX packets:29527 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:841385910 (802.4 MiB)  TX bytes:5126494 (4.8 MiB)
          Interrupt:169
```

```
lo        Link encap:Local Loopback
          inet addr:127.0.0.1  Mask:255.0.0.0
          UP LOOPBACK RUNNING  MTU:16436  Metric:1
          RX packets:54 errors:0 dropped:0 overruns:0 frame:0
```

TX packets:54 errors:0 dropped:0 overruns:0 carrier:0
collisions:0 txqueuelen:0
RX bytes:4452 (4.3 KiB) TX bytes:4452 (4.3 KiB)

Running script NIS
Wed Apr 9 17:40:04 PDT 2008
Binding to the NIS domain: [60G[[0;32mOK[0;39m]
Listening for an NIS domain server.
ypbind (pid 22277) is running...
.....
xmeta:\$1\$zKiVeV/P\$CaMDMX27mXoIc7GEeo/M5/:501:501::/home/xmeta:/bin/bash
tom:\$1\$TW5X1inw\$W6b6D0fHogPjUSUxFI6Nt1:510:514:Tom Jones:/home/tom:/bin/bash
nagraj:\$1\$5mQwb571\$9/t8hMnIb/gEypVicQUH./:500:500:Nagraj
Alur:/home/nagraj:/bin/bash
dsadm:\$1\$XGsV0dFD\$cOTJlVh4w2UNzZWlBXAlw0:507:512::/home/dsadm:/bin/bash
loadl:\$1\$QKw1NMvP\$RbXf2vRBE0srzHECG7UsU0:516:519::/home/loadl:/bin/bash
mqm:!!:515:518::/var/mqm:/bin/bash
wasadmin:\$1\$f.KxX8S4\$6skaDDxhK0sP3uThcLucP.:505:505::/home/wasadmin:/bin/bash
db2fenc1:\$1\$82zXnPAC\$8VY9A02RARIoXl4/qNMEf/:503:510::/home/db2fenc1:/bin/bash
admin:\$1\$AVem7Z3M\$AVOYBiVQL33KlnhIJ8YQY1:508:508::/home/admin:/bin/bash
dasusr1:\$1\$UbdFBWeV\$3G254CQqAINK1sE98.1aa1:504:511::/home/dasusr1:/bin/bash
fred:!!:517:520::/home/fred:/bin/bash
per:\$1\$zKckxjCA\$lgGbKT5XR87Sd1YC1eBSj0:513:517:Per Hansen:/home/per:/bin/bash
db2inst1:\$1\$CCC0caT4\$mjX.oS30dSONlqI0eOsAh.:502:509::/home/db2inst1:/bin/bash
.....
Wed Apr 9 17:42:04 PDT 2008
Running script NFS
Wed Apr 9 17:42:04 PDT 2008
Filesystem 1K-blocks Used Available Use% Mounted on
/dev/VolGroup00/LogVol100
574536368 2566932 542784620 1% /
/dev/sda1 101086 13643 82224 15% /boot
192.168.1.101:/tftpboot 574536384 49693600 495657952 10% /mnt/tmp
mkdir: created directory ~/ds_overview'
mkdir: created directory ~/opt/IBM'
Filesystem 1K-blocks Used Available Use% Mounted on
/dev/VolGroup00/LogVol100
574536368 2566952 542784600 1% /
/dev/sda1 101086 13643 82224 15% /boot
192.168.1.101:/tftpboot 574536384 49693600 495657952 10% /mnt/tmp
192.168.1.77:/opt/IBM 98435488 12986688 80448576 14% /opt/IBM
192.168.1.77:/ds_overview 98435488 95584 93339680 1% /ds_overview
192.168.1.77:/home 196871008 32072416 154798144 18% /home
Wed Apr 9 17:42:04 PDT 2008
Running script LoadL

Wed Apr 9 17:42:04 PDT 2008
192.168.1.101orionpvt.itsosj.sanjose.ibm.com orionpvt
9.43.86.102phoenix.itsosj.sanjose.ibm.com phoenix
9.43.86.77kazan.itsosj.sanjose.ibm.com kazan
192.168.1.134luxorpvt.itsosj.sanjose.ibm.com luxorpvt
9.43.86.76nile.itsosj.sanjose.ibm.com nile
192.168.1.134luxorpvt.itsosj.sanjose.ibm.com luxorpvt
192.168.1.76nilepvt.itsosj.sanjose.ibm.com nilepvt
192.168.1.77kazanpvt.itsosj.sanjose.ibm.com kazanpvt
127.0.0.1orion.itsosj.sanjose.ibm.com orion localhost.localdomain localhost
9.43.86.101orion.itsosj.sanjose.ibm.com orion
192.168.1.102phoenixpvt.itsosj.sanjose.ibm.com phoenixpvt
192.168.1.77kazanpvt.itsosj.sanjose.ibm.com kazanpvt
127.0.0.1orion.itsosj.sanjose.ibm.com orion localhost.localdomain localhost
9.43.86.102phoenix.itsosj.sanjose.ibm.com phoenix
192.168.1.76nilepvt.itsosj.sanjose.ibm.com nilepvt
9.43.86.134luxor.itsosj.sanjose.ibm.com luxor
192.168.1.101orionpvt.itsosj.sanjose.ibm.com orionpvt
9.43.86.101orion.itsosj.sanjose.ibm.com orion
9.43.86.77kazan.itsosj.sanjose.ibm.com kazan
192.168.1.102phoenixpvt.itsosj.sanjose.ibm.com phoenixpvt
9.43.86.76nile.itsosj.sanjose.ibm.com nile
9.43.86.134luxor.itsosj.sanjose.ibm.com luxor
IBMJava2-JRE-1.4.2-0.0.i386.rpm
LoadL-full-RH3-X86-3.3.2.0-0.i386.rpm
LoadL-full-RH3-X86_64-3.3.2.0-0.x86_64.rpm
LoadL-full-RH4-X86-3.3.2.0-0.i386.rpm
LoadL-full-RH4-X86_64-3.3.2.0-0.x86_64.rpm
LoadL-full-SLES9-X86-3.3.2.0-0.i386.rpm
LoadL-full-SLES9-X86_64-3.3.2.0-0.x86_64.rpm
LoadL-full-lib-RH4-X86-3.3.2.0-0.i386.rpm
LoadL-full-lib-SLES9-X86-3.3.2.0-0.i386.rpm
LoadL-full-license-RH3-X86-3.3.2.0-0.i386.rpm
LoadL-full-license-RH3-X86_64-3.3.2.0-0.x86_64.rpm
LoadL-full-license-RH4-X86-3.3.2.0-0.i386.rpm
LoadL-full-license-RH4-X86_64-3.3.2.0-0.x86_64.rpm
LoadL-full-license-SLES9-X86-3.3.2.0-0.i386.rpm
LoadL-full-license-SLES9-X86_64-3.3.2.0-0.x86_64.rpm
LoadL-so-RH3-X86-3.3.2.0-0.i386.rpm
LoadL-so-RH3-X86_64-3.3.2.0-0.x86_64.rpm
LoadL-so-RH4-X86-3.3.2.0-0.i386.rpm
LoadL-so-RH4-X86_64-3.3.2.0-0.x86_64.rpm
LoadL-so-SLES9-X86-3.3.2.0-0.i386.rpm
LoadL-so-SLES9-X86_64-3.3.2.0-0.x86_64.rpm
LoadL-so-license-RH3-X86-3.3.2.0-0.i386.rpm
LoadL-so-license-RH3-X86_64-3.3.2.0-0.x86_64.rpm
LoadL-so-license-RH4-X86-3.3.2.0-0.i386.rpm
LoadL-so-license-RH4-X86_64-3.3.2.0-0.x86_64.rpm
LoadL-so-license-SLES9-X86-3.3.2.0-0.i386.rpm

```

LoadL-so-license-SLES9-X86_64-3.3.2.0-0.x86_64.rpm
LoadL.README
LoadL.README.RH3_X86_64_AFS
LoadL.README.RH3_X86_AFS
Preparing... #####
LoadL-full-license-RH4-X86 #####
Installing /tmp/load1/IBMJava2-JRE-1.4.2-0.0.i386.rpm at /opt/ibm11
Preparing... #####
IBMJava2-JRE #####
Running the IBM License Acceptance Program...
Uninstalling the package IBMJava2-JRE-1.4.2-0.0 installed at /opt/ibm11.
Installing the RPM: /tmp/load1/LoadL-full-RH4-X86-3.3.2.0-0.i386.rpm
Preparing... #####
LoadL-full-RH4-X86 #####
The rpm post-install phase may take a few minutes. Please be patient...
/opt/ibm11/LoadL/full/bin/llinit: creating directory "/var/load1/spool".
/opt/ibm11/LoadL/full/bin/llinit: creating directory "/var/load1/log".
/opt/ibm11/LoadL/full/bin/llinit: creating directory "/var/load1/execute".
/opt/ibm11/LoadL/full/bin/llinit: set permission "700" on "/var/load1/spool".
/opt/ibm11/LoadL/full/bin/llinit: set permission "775" on "/var/load1/log".
/opt/ibm11/LoadL/full/bin/llinit: set permission "1777" on
"/var/load1/execute".
/opt/ibm11/LoadL/full/bin/llinit: ==> file /home/load1/LoadL_admin already
exists.
/opt/ibm11/LoadL/full/bin/llinit: ==> No copy will be done.
/opt/ibm11/LoadL/full/bin/llinit: ==> file /home/load1/LoadL_config already
exists.
/opt/ibm11/LoadL/full/bin/llinit: ==> No copy will be done.
/opt/ibm11/LoadL/full/bin/llinit: creating file
"/var/load1/LoadL_config.local".

/opt/ibm11/LoadL/full/bin/llinit: ==> The keyword "RELEASEDIR" was not updated.
/opt/ibm11/LoadL/full/bin/llinit: ==> The Central Manager machine stanza was
not updated.
/opt/ibm11/LoadL/full/bin/llinit: ==> directory /home/load1/spool already
exists.
/opt/ibm11/LoadL/full/bin/llinit: ==> directory /home/load1/log already exists.
/opt/ibm11/LoadL/full/bin/llinit: ==> directory /home/load1/execute already
exists.
/opt/ibm11/LoadL/full/bin/llinit: ==> symbolic link
"/home/load1/LoadL_config.local -> /var/load1/LoadL_config.local" already
exists.
/opt/ibm11/LoadL/full/bin/llinit: ==> directory /home/load1/bin already exists.
/opt/ibm11/LoadL/full/bin/llinit: ==> directory /home/load1/lib already exists.
/opt/ibm11/LoadL/full/bin/llinit: ==> directory /home/load1/man already exists.
/opt/ibm11/LoadL/full/bin/llinit: ==> directory /home/load1/samples already
exists.
/opt/ibm11/LoadL/full/bin/llinit: ==> directory /home/load1/include already
exists.

```

```

/opt/ibmll/LoadL/full/bin/llinit: program complete.
Wed Apr 9 17:42:25 PDT 2008
Running script ganglia_gmond
Wed Apr 9 17:42:25 PDT 2008
Preparing... #####
ganglia-gmond #####
Starting GANGLIA gmond: [ OK ]
Shutting down GANGLIA gmond: [ OK ]
Starting GANGLIA gmond: [ OK ]
Wed Apr 9 17:42:26 PDT 2008
Wed Apr 9 17:42:26 PDT 2008

```

- Example 2-102 on page 129 shows the contents of the Loadl_config.local file as installed on the compute node.

It shows a maximum of one job that can be started on this compute node (MAX_STARTERS= 1).

A number of job classes can execute on this compute node:

- small(8)
- medium(5)
- large(2)
- No_Class(1)

The number in the parenthesis is the maximum number of jobs relating to this class that may execute on this compute node.

Note: MAX_STARTERS determines the actual number of jobs of all classes that may run concurrently on this compute node. With MAX_STARTERS= 1 only one job of either the small class, the medium class, the large class, or the No_Class class may be running. The number in the parenthesis for each class is overridden by the MAX_STARTERS value.

For example, if MAX_STARTERS were set to 6, then at any instant for the values configured in Example 2-102, there can be no more than six class small jobs, five class medium jobs, two class large jobs, or one class No_Class jobs running on the compute node. The permitted jobs per class running concurrently together cannot exceed six.

The considerations in configuring the MAX_STARTERS parameter (the number of concurrent jobs) is not described here. Carefully evaluate (through regression testing) the impact of increasing the MAX_STARTERS value from the recommended value of 1.

Example 2-102 LoadL_config.local on the compute node

```
# # file: LoadL_config.local
# Statements included here override on a PER MACHINE BASIS the
# statements in
# LoadL_config and may be used to tailor each machine individually.
# See samples in your RELEASEDIR/samples directory
#
START_DAEMONS= TRUE
MAX_STARTERS= 1

# Alternative ways of specifying the CLASS keyword
# the following is the old-style specification
#
#CLASS = { "small" "small" "small" "small" "small" "small" "small"
"small" "medium" "medium" "medium" "medium" "medium" "large" "large" }
# while the following is a newer, more concise specification
#CLASS = small(8) medium(5) large(2)
CLASS = small(8) medium(5) large(2) No_Class(1)
```

Attention: This process must be repeated for the remaining two compute nodes with IP addresses 192.168.1.102 and 192.168.1.134. This is not shown here.

At the end of the installation and configuration of all the compute nodes, the contents of some of the key files was as follows:

- Example 2-103 shows the /etc/hosts file contents on the front end node.

Example 2-103 /etc/hosts file

```
# Do not remove the following line, or various programs
# that require network functionality will fail.
127.0.0.1    localhost.localdomain localhost
192.168.1.101 orionpvt.itsosj.sanjose.ibm.com orionpvt
192.168.1.134 luxorpvt.itsosj.sanjose.ibm.com luxorpvt
192.168.1.76 nilepvt.itsosj.sanjose.ibm.com nilepvt
192.168.1.77 kazanpvt.itsosj.sanjose.ibm.com kazanpvt
9.43.86.101  orion.itsosj.sanjose.ibm.com orion
9.43.86.134  luxor.itsosj.sanjose.ibm.com luxor
9.43.86.76   nile.itsosj.sanjose.ibm.com nile
9.43.86.77   kazan.itsosj.sanjose.ibm.com kazan
```

- Example 2-104 shows the /etc/dhcpd.conf file contents.

The three compute nodes added to this file's contents is what ensures that the next PXE client boot process does not reuse the assigned IP addresses.

Example 2-104 /etc/dhcpd.conf file

```
ddns-update-style interim;
ignore client-updates;

allow booting;
allow bootp;

subnet 192.168.1.0 netmask 255.255.255.0 {

# --- default gateway
    option routers                192.168.1.101;
    option subnet-mask            255.255.255.0;

    option nis-domain             "IS.redbook.grid";
    option domain-name            "itsosj.sanjose.ibm.com";
    option domain-name-servers    192.168.1.101;

    option time-offset            -18000; # Eastern
Standard Time

    range dynamic-bootp 192.168.1.102 192.168.1.103;
    default-lease-time 21600;
    max-lease-time 43200;

    next-server 192.168.1.101;
    filename "/linux-install/pxelinux.0";
    host taruspvt.itsosj.sanjose.ibm.com {
        hardware ethernet 00:14:5e:a4:e4:c7;
        fixed-address 192.168.1.103;
        option host-name "taruspvt.itsosj.sanjose.ibm.com";
    }
    host phoenixpvt.itsosj.sanjose.ibm.com {
        hardware ethernet 00:14:5e:ac:36:32;
        fixed-address 192.168.1.102;
        option host-name "phoenixpvt.itsosj.sanjose.ibm.com";
    }
    host luxorpvt.itsosj.sanjose.ibm.com {
        hardware ethernet 00:02:55:AC:86:D2;
        fixed-address 192.168.1.134;
```



```
        option host-name "luxorprt.itsosj.sanjose.ibm.com";  
    }  
}
```

2.5.6 Step 1f: Installing and configuring Grid Enablement Toolkit

The Grid Enablement Toolkit is a set of processes and templates that allows you to execute jobs in the grid environment infrastructure built using the BYOG toolkit. In a grid implementation, the Grid Enablement Toolkit replaces the static configuration files normally defined for a cluster with dynamic configuration files created at runtime.

In this step, we install the Grid Enablement Toolkit, modify the contents of the config and dsenv files, and run a test script to verify that the grid environment is configured and operating correctly.

Installing the toolkit

Log in as user dsadm, change the current working directory to \$DSHOME/./PXEngine, and perform the following steps:

1. Extract the toolkit installation files using the tar command as shown in Example 2-105.

Note: Needs to be installed with user dsadm with DataStage running.

Example 2-105 Extract Grid Enablement Toolkit

```
[dsadm@luxor PXEngine]$ tar -xvf /tmp/grid_enabled.4.0.1.tar  
grid_enabled.4.0.1/  
grid_enabled.4.0.1/GridSampleJobs.dsx  
grid_enabled.4.0.1/Media  
grid_enabled.4.0.1/install.sh  
grid_enabled.4.0.1/README.txt
```

2. Execute the install.sh script and provide the highlighted responses shown in Example 2-106 on page 132.

The highlighted comments at the end of the execution specify the next actions to take, such as restarting DataStage, sourcing the dsenv file, modifying the grid_config.apt file, and running the test.sh script.

Example 2-106 Install the Grid Enablement Toolkit

```
[dsadm@orion grid_enabled.4.0.1]$ ./install.sh
```

```
install - Release 4.0.1 March 18, 2008
```

```
Installation Server Type
```

```
1) Information Server Install
```

```
2) 7.x install
```

```
Server type: 1
```

```
Resource Manager selection List:
```

```
Condor, DataSynapse, LoadLeveler, LSF, PBSPro, SGE, Torque, or SMP: LoadLeveler
```

```
Path to Resource Manager commands: /home/loadl/bin
```

```
Grid JOBDIR (work directory that must be shared on all nodes)
```

```
Directory (full path): /home/dsadm/grid_jobdir
```

```
Maximum number of nodes per job (global): 3
```

```
Enter a space separated list of valid Classes for run time validation
```

```
List of valid LoadLeveler Classes: small medium large No_Class
```

```
Enter a default Class for running jobs: medium
```

```
Use system defaults for all projects?
```

```
default Classes are: small medium large No_Class
```

```
default grid JOBDIR is: /home/dsadm/grid_jobdir
```

```
Use defaults (Y/N): Y
```

```
Adding Grid parameters to DataStage Projects
```

```
Processing /opt/IBM/InformationServer/Server/Projects/ANALYZERPROJECT
```

```
Processing /opt/IBM/InformationServer/Server/Projects/AlokQsResidencyProject
```

```
Processing /opt/IBM/InformationServer/Server/Projects/BarryRosenProject
```

```
Processing /opt/IBM/InformationServer/Server/Projects/CELSOT
```

```
Processing /opt/IBM/InformationServer/Server/Projects/CELSOT_REVIEW
```

```
Processing /opt/IBM/InformationServer/Server/Projects/CarmenTestDS
```

```
Processing /opt/IBM/InformationServer/Server/Projects/DS_Overview
```

```
Processing /opt/IBM/InformationServer/Server/Projects/Denis
```

```
Processing /opt/IBM/InformationServer/Server/Projects/IS_Grid
```

```
Processing /opt/IBM/InformationServer/Server/Projects/NAGRAJDS
```

```
Processing /opt/IBM/InformationServer/Server/Projects/PROJQSSAMP
```

```
Processing /opt/IBM/InformationServer/Server/Projects/QSBusinessScenario
```

```
Processing /opt/IBM/InformationServer/Server/Projects/QSTestProject
```

```
Processing /opt/IBM/InformationServer/Server/Projects/Sachiko
```

```
Processing /opt/IBM/InformationServer/Server/Projects/TorbenSkov
```

```
Processing /opt/IBM/InformationServer/Server/Projects/jnielsen
```

```
Processing /opt/IBM/InformationServer/Server/Projects/skovpart1
```

```
Processing /opt/IBM/InformationServer/Server/DSEngine/../../Template
```

DataStage needs to be restarted.

Prior to testing, you must source the dsenv file

*Example: . /opt/IBM/InformationServer/Server/DSEngine/dsenv
to get an updated PATH prior to running the test script*

*The default grid_config.apt file is created as a global template file.
You can copy this into a project directory to create a default for a given
project (name must rename the same). You can create additional copies of
a template file and make modifications for specific jobs and then use the
\$APT_GRID_CONFIG parameter to identify the template for the job.*

Testing: run (test.sh) script and validate that it returns correctly

Note: you can not run the test.sh script as root

Restarting DataStage

Perform the following steps to restart DataStage:

1. Stop the current running instance using the following command:
`/opt/IBM/InformationServer/Server/PXEngine/bin/uv -admin -stop`
2. Validate that all network connections are ended using the following command:
`netstat -a|grep dsrpc`
The output must be empty.
3. Start the DataStage server using the following command:
`/opt/IBM/InformationServer/Server/PXEngine/bin/uv -admin -start`

Review key files of interest

After installation, the following files are of interest:

- `grid_global_values`

Example 2-107 shows the contents of the `grid_global_values` file in the
`/opt/IBM/InformationServer/Server/PXEngine/grid_enabled.4.0.1/` directory
which lists the default values.

Example 2-107 grid_global_values file content

```
# Grid Engine, review supported Resource Managers for valid values
Engine=LoadLeveler
# Add this to the PATH, SOURCE this script and/or SET and export this value
PATH=/home/loadl/bin
SOURCE=
SET=
# Maximum nodes a job can have
```

MaxNodesPerJob=3

ViewDataNodes = Max number of nodes for view data

ViewDataNodes=3

ViewDataPartitions = Number of partitions per node for view data

ViewDataPartitions=4

List of valid Queues/classes that can be used for job submission

ValidQueues=small medium large No_Class

If APT_GRID_QUEUE is not defined, use this value (can be blank for no queue)

DefaultQueue=medium

List of aliases for the conductor node. Prevents using static config files

ConductorAKA=localhost

Default Grid job directory (work file location)

JOBDIR=/home/dsadm/grid_jobdir

Pre 4.0 release format for node names (If yes node names are X_Y otherwise Z

where X is physical server, Y = partition or Z = simple format

OldFormatStyle=No

Allow jobs to run on Conductor node only (Project overrides)

ConductorOnly=No

► dsenv file

Example 2-108 shows the partial contents of the dsenv file with the updated path statements.

Example 2-108 dsenv file: Modified contents

.....

For virtual host name being kazanpvt.itsosj.sanjose.ibm.com

APT_PM_CONDUCTOR_HOSTNAME=kazanpvt.itsosj.sanjose.ibm.com

export APT_PM_CONDUCTOR_HOSTNAME

For Patch ECASE: 124940

DS_HOSTNAME_ALIAS=Kazan.itsosj.sanjose.ibm.com

export DS_HOSTNAME_ALIAS

.....

GRIDHOME=/opt/IBM/InformationServer/Server/PXEngine/grid_enabled.4.0
.1

JAVAPATH=/opt/IBM/InformationServer/ASBNode/apps/jre/bin

export GRIDHOME JAVAPATH

PATH=\$GRIDHOME:\$JAVAPATH:\$PATH

. \$GRIDHOME/gridEnv.sh

► **grid_config.apt** file

Example 2-109 shows the contents of the `grid_config.apt` file in the `/opt/IBM/InformationServer/Server/PXEngine/grid_enabled.4.0.1/` directory.

The `fastname` must be modified to `kazanpvt.itsosj.sanjose.ibm.com` (node name corresponding to the private network) because it must match the `APT_PM_CONDUCTOR_HOSTNAME` environment variable in the `dsenv` file (Example 2-108 on page 134). This determines the node name that section leaders on the compute node use to connect back to the conductor.

Example 2-109 Grid_config.apt file content

```
{
  node "Conductor"
  {
    fastname "kazanpvt.itsosj.sanjose.ibm.com"
    pools "conductor"
    resource disk "/opt/IBM/InformationServer/Server/Datasets"
{pools ""}
    resource scratchdisk "/tmp" {pools ""}
  }
  node "$$Compute"
  {
    fastname "$$fastname"
    pools ""
    resource disk "/opt/IBM/InformationServer/Server/Datasets"
{pools ""}
    resource scratchdisk "/tmp" {pools ""}
  }
}
```

► **Node_table**

A `Node_table` translates the `fastnames` from one host name to another. The intent is to control on which network the DataStage job runs. We have defined all compute node private network IP addresses as 192.168.1.x, and with host names ending in “pvt” as follows:

- `tarus.itsosj.sanjose.ibm.com`: public name
 `taruspvt.itsosj.sanjose.ibm.com`: private name
- `phoenix.itsosj.sanjose.ibm.com`: public name
 `phoenixpvt.itsosj.sanjose.ibm.com`: private name
- `luxor.itsosj.sanjose.ibm.com`: public name
 `luxorpvt.itsosj.sanjose.ibm.com`: private name

Example 2-110 shows the original contents of the Node table. Example 2-111 shows the modified contents of the Node table to perform the translation from a public network to the private network.

Example 2-110 Node table: Original contents

```
s/compute-0-0.local/compute-0-0/g
s/compute-0-1.local/compute-0-1/g
s/compute-0-2.local/compute-0-2/g
```

Example 2-111 Node table: Modified contents

```
s/luxor.itsosj.sanjose.ibm.com/luxorpvt.itsosj.sanjose.ibm.com/g
s/tarus.itsosj.sanjose.ibm.com/taruspvt.itsosj.sanjose.ibm.com/g
s/phoenix.itsosj.sanjose.ibm.com/phoenixpvt.itsosj.sanjose.ibm.com/g
```

Source the dsenv file

For the changes made to the dsenv file to take effect, you must source it as shown in Example 2-111.

Example 2-112 Source the dsenv file

```
# . /opt/IBM/InformationServer/Server/DSEngine/dsenv
```

Executing the test.sh script

To test the installation of the grid_enabled components, the test.sh script generated in the /opt/IBM/InformationServer/Server/PXEngine/grid_enabled.4.0.1/ directory during the install can be run as shown in Example 2-112. It performs the following tasks:

1. Submits a request to the Resource Manager (LoadLeveler in our case) for the defined nodes.
2. Generates a configuration file.
3. Runs a simple osh command with output to peek.

The output of the test.sh script are the contents of the APT configuration file and a few rows from an osh script. If the test script does not return within a few seconds, something is not configured correctly. The message “Step execution finished with status = OK.” indicates successful execution of the test.sh script.

Results of the execution of this script show the job belonging to the No_Class LoadLeveler queue, and phoenixpvt.itsosj.sanjose.ibm.com as the compute

node. It also identifies the dynamically created configuration file as being /home/dsadm/grid_jobdir/testjob_26817.config.apt.

Note: test.sh runs with a default of one compute node and one partition. The following lines can be modified to suit to test with more nodes and partitions:

- ▶ line 10 APT_GRID_COMPUTENODES=1
- ▶ line 11 APT_GRID_PARTITIONS=1

A number of temporary files are created in the /home/dsadm/grid_jobdir while a job is running and after it has completed. This is shown in Example 2-114 on page 139 and Example 2-115 on page 139. The information shown here does not relate to the execution of the test.sh script. but jobs run later.

Example 2-113 Execute test.sh script

```
[dsadm@orion grid_enabled.4.0.1]$ ./test.sh
You need to source the /opt/IBM/InformationServer/Server/DSEngine/dsenv file first
[dsadm@orion grid_enabled.4.0.1]$ . /opt/IBM/InformationServer/Server/DSEngine/dsenv
[dsadm@orion grid_enabled.4.0.1]$ ./test.sh
If everything is setup correctly this script will end shortly.
```

However, if this script does not return within a few seconds, then you need to perform the following:

1. issue qstat, bstat or llstatus command to see if a job was submitted
2. If no job is queue'd then make sure the qsub/llsubmit/ds_submit or bsub command is in your path and try again
3. Validate that java is in the path on the compute node
This is also required on the front-end node
4. Validate that the parameters are defined correctly and resources are available (APT_PM_CONDUCTOR_HOSTNAME is set correctly in test.sh)

```
<Dynamic_grid.sh> LoadLeveler output files located in /home/dsadm/grid_jobdir
nohup: appending output to `nohup.out'
<Dynamic_grid.sh> Waiting for Release from Queue, Jobname: testjob_26817
<Dynamic_grid.sh> Dynamic Configuration Release 4.0.1 March 19, 2008
<Dynamic_grid.sh> Released from LoadLeveler Queue medium
<Dynamic_grid.sh> Compute node(s): phoenixpvt.itsosj.sanjose.ibm.com
<Dynamic_gird.sh> SEQFILE Host(s): phoenixpvt.itsosj.sanjose.ibm.com:
phoenixpvt.itsosj.sanjose.ibm.com:
{
    node "Conductor"
    {
        fastname "kazanpvt.itsosj.sanjose.ibm.com"
        pools "conductor"
        resource disk "/opt/IBM/InformationServer/Server/Datasets" {pools ""}
        resource scratchdisk "/tmp" {pools ""}
    }
    node "Compute1"
    {
```

```

        fastname "phoenixpvt.itsosj.sanjose.ibm.com"
        pools ""
        resource disk "/opt/IBM/InformationServer/Server/Datasets" {pools ""}
        resource scratchdisk "/tmp" {pools ""}
    }
}

```

**You can view the configuration file created by viewing
/home/dsadm/grid_jobdir/testjob_26817.config.apt**

APT_GRID_SEQFILE_HOST is set to: phoenixpvt.itsosj.sanjose.ibm.com:
APT_GRID_SEQFILE_HOST2 is set to: phoenixpvt.itsosj.sanjose.ibm.com:

Now testing the osh grid enablement...

##I XXXXXB 000001 <osh_conductor> Checking Authorization...

##I XXXXXF 000001 <osh_conductor> Authorized to proceed.

##I IIS-DSEE-TFCN-00001 18:22:08(000) <main_program>

IBM WebSphere DataStage Enterprise Edition 8.0.1.4668

Copyright (c) 2001, 2005-2007 IBM Corporation. All rights reserved

##I IIS-DSEE-TOSH-00002 18:22:08(001) <main_program> orchgeneral: loaded

##I IIS-DSEE-TOSH-00002 18:22:08(002) <main_program> orchsort: loaded

##I IIS-DSEE-TOSH-00002 18:22:08(003) <main_program> orchstats: loaded

##I IIS-DSEE-TFSC-00001 18:22:08(007) <main_program> **APT configuration file:**

/home/dsadm/grid_jobdir/testjob_26817.config.apt

##I IIS-DSEE-TOPK-00003 18:21:52(000) <peek,0> 0 aa

##I IIS-DSEE-TOPK-00003 18:21:52(001) <peek,0> 1 bbbbbb

##I IIS-DSEE-TOPK-00003 18:21:52(002) <peek,0> 2 ccccc

##I IIS-DSEE-TOPK-00003 18:21:52(003) <peek,0> 3 ddddd

##I IIS-DSEE-TOPK-00003 18:21:52(004) <peek,0> 4 eeeee

##I IIS-DSEE-TOPK-00003 18:21:52(005) <peek,0> 5 ff

##I IIS-DSEE-TOPK-00003 18:21:52(006) <peek,0> 6 ggg

##I IIS-DSEE-TOPK-00003 18:21:52(007) <peek,0> 7 hhhh

##I IIS-DSEE-TOPK-00003 18:21:52(008) <peek,0> 8 iii

##I IIS-DSEE-TOPK-00003 18:21:52(009) <peek,0> 9 jjjjj

##I IIS-DSEE-TFSC-00010 18:22:08(009) <main_program> **Step execution finished with status = OK.**

##I IIS-DSEE-TCOS-00026 18:22:08(010) <main_program> Startup time, 0:00; production run time, 0:00.

Done - You should see generated output in peeks from the framework
and the message "Step execution finished with status = OK"

Example 2-114 Sample /home/dsadm/grid_jobdir contents while a job is running

```
.....
-rw-r--r-- 1 dsadm dstage 315 Apr 18 18:18 testjob.time
-rwxr----- 1 dsadm dstage 440 Apr 18 18:21 testjob_26817.wait.sh
-rw-rw-r-- 1 dsadm dstage 0 Apr 18 18:21 testjob_26817.stdout
-rw-rw-r-- 1 dsadm dstage 0 Apr 18 18:21 testjob_26817.stderr
-rwxr----- 1 dsadm dstage 2082 Apr 18 18:21 testjob_26817.sh
-rw-r--r-- 1 dsadm dstage 167 Apr 18 18:22 testjob_26817_start
-rw-r--r-- 1 dsadm dstage 98 Apr 18 18:22 testjob_26817.nodeFile
-rw-r--r-- 1 dsadm dstage 1362 Apr 18 18:22 testjob_26817.config.apt
.....
```

Example 2-115 Sample /home/dsadm/grid_jobdir contents after a job has finished

```
.....
-rw-r--r-- 1 dsadm dstage 315 Apr 18 18:18 testjob_26817.time
-rw-rw-r-- 1 dsadm dstage 0 Apr 18 18:22 testjob_26817_end
.....
```

Toolkit environment variables

For 8.x applications, the toolkit environment variables are set using the DSParams file which is updated during installation. Within IBM Information Server, the project defaults are used for all jobs including Information Analyzer. Specific jobs must override defaults in order to change how the components react.

Each project within IBM Information Server is modified to include the grid-related environment variables shown in Figure 2-15 on page 140. These are defined as project level defaults and can be added or changed at the job level. There is a new tab (highlighted in Figure 2-15 on page 140) to identify all of the grid parameters.

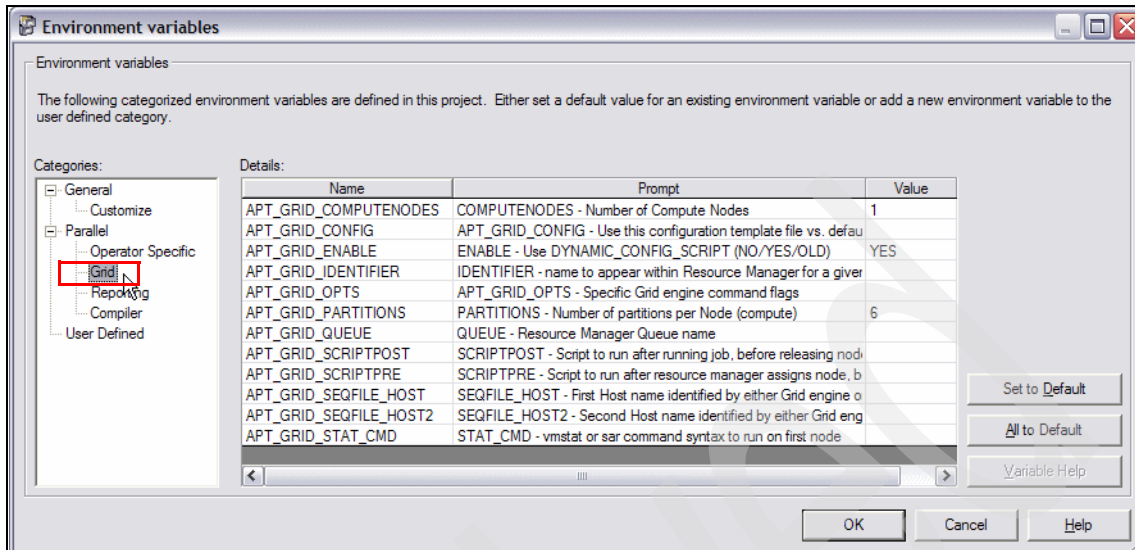


Figure 2-15 Grid environment variables

A brief summary of the grid environment variables follows:

► **\$APT_GRID_COMPUTENODES**

Specifies the number of compute nodes (machines) required for this job. The default 1.

► **\$APT_GRID_CONFIG**

Specifies the path name to the template \$APT_CONFIG_FILE file that must be used for a given job run.

The default is to use the project's grid_config.appt configuration file. If such a file does not exist in the project directory, then the global grid_config.appt configuration file is used.

Setting this parameter defines the template file to use for this job run.

► **\$APT_GRID_ENABLE**

Has two possible values:

- (Y)es indicates that a new configuration file will be created dynamically based on other parameters such as \$APT_GRID_COMPUTENODES and \$APT_GRID_PARTITIONS.
- (N)o indicates that a defined configuration file is available and will be used. This is the default.

► \$APT_GRID_IDENTIFIER

Setting this overrides the actual job name when submitted to the resource manager.

► \$APT_GRID_OPTS

Additional options may be supplied to the Grid engine to request resources.

Node resource parameters must not be included here, as these are requested using the environment variable \$APT_GRID_COMPUTENODES.

These parameters are based on a specific resource manager and are not translated.

► \$APT_GRID_PARTITIONS

Specifies the number of partitions for each compute node. The default is 1 partition per compute node.

Note: This value can be set to a higher value if the nodes (machines) have additional CPU cycles.

► \$APT_GRID_QUEUE

The name of the resource manager queue to which a job is submitted. The default is blank or undefined.

Queues are used to separate jobs on the basis of priority:

- High
- Medium
- Low
- Special processing

Each queue typically has different resources available to run jobs.

Note: If the value is blank and there is a default queue defined for either the project or at the global level, that value is used when jobs are submitted to the resource manager.

► \$APT_GRID_SCRIPTPOST

Name of an executable script that gets run after the DataStage job finishes processing data, but before the job completes. The script is run on the node that the configuration file was created on.

► **\$APT_GRID_SCRIPTPRE**

Name of a script that can be executed after the resource manager has allocated the desired nodes, but before the actual DataStage job starts processing. This script is run on the node that creates the configuration file.

► **\$APT_GRID_SEQFILE_HOST**

Has the host name set by the `Dynamic_grid.sh` based on the first node returned by the resource manger. It is used as a job parameter in DataStage and QualityStage jobs to provide sequential file I/O on files.

Example usage when defining path name:

```
#$APT_GRID_SEQFILE_HOST#/home/dsadm/Ascential/DataStage/Datasets/StoreList.txt
```

When used as shown, “hostname:” is returned. The colon is included in the value returned.

► **\$APT_GRID_SEQFILE_HOST2**

Has the host name set by `Dynamic_grid.sh` based on the second node returned by the resource manger. Similar to `$APT_GRID_SEQFILE_HOST`. It has the first host name if only one host name is returned by the resource manager.

► **\$APT_GRID_STAT_CMD**

Setting this parameter to an operating system level monitoring command allows the output to be captured on the first compute node returned by the resource manager.

This parameter is useful when trying to determine how the first compute node is performing during a job run.

For example, the command `sar 5 1000` runs `sar` every five seconds for 1000 times, or until the job ends (the last step before the job ends on the compute kills this process). The suffix of the file containing the output of this command is `statcmd`. The file is located in the `<JOBDIR>` as the `<JOBNAME>_PID.statcmd`.

2.5.7 Step 1g: Configuring the standby node

As part of our HA solution, we have a dedicated standby node. When the front end node fails, the standby node takes over the role of front end node. Therefore, at failover, the standby node must have access to all the components of the front end node, including the following components:

- ▶ IBM Information Server
- ▶ Workload Scheduler LoadLeveler
- ▶ Ganglia
- ▶ NFS
- ▶ NIS
- ▶ The virtual IP address and its corresponding domain name

To achieve this, the following actions must be taken on the standby node:

- ▶ IBM Workload Scheduler LoadLeveler must be installed and configured on this node, but it must not be started on this node. The procedure is similar to the one described in “Installing and configuring Workload Scheduler LoadLeveler” on page 102 and as such, is not repeated here.
- ▶ The following code libraries and configuration files must be shared between the primary front end node and the standby node:
 - IBM Information Server
 - Workload Scheduler LoadLeveler
 - NFS
 - NIS

Also included are the code libraries and configuration files of IBM WebSphere MQ product. These code libraries and configuration files are placed in the following file systems on the shared SAN:

- /opt/IBM
 - /home
 - /ds_overview
 - /opt/mqm
- ▶ In order to see the charts on Ganglia, you must disable SELINUX on the standby node and set it to permissive by issuing the following command:
`setenforce 0`
- ▶ Ganglia must be installed and configured (Example 2-116 on page 144) without starting it because its code libraries and configuration files were not installed on the shared SAN. The Ganglia package installed the libraries in directories that were not part of the SAN shared drive.

Example 2-116 Install ganglia

```
# rpm -ivh ganglia-gmond-3.0.2-1.i386.rpm
Preparing... ##### [100%]
1:ganglia-gmond ##### [100%]
Starting GANGLIA gmond: [ OK ]

# rpm -ivh ganglia-gmetad-3.0.2-1.i386.rpm
Preparing... ##### [100%]
1:ganglia-gmetad ##### [100%]
Starting GANGLIA gmetad: [ OK ]

# rpm -ivh ganglia-web-3.0.2-1.noarch.rpm
Preparing... ##### [100%]
1:ganglia-web ##### [100%]

# rpm -ivh rrdtool-1.0.50-2.1.el3.rf.i386.rpm
warning: rrdtool-1.0.50-2.1.el3.rf.i386.rpm: V3 DSA signature: NOKEY, key ID 6b8d79e6
Preparing... ##### [100%]
1:rrdtool ##### [100%]
```

Configure ganglia by editing the cluster name in the gmond configuration file. Change the cluster name from name = “unspecified” to name = “Data Integration”, but do not start the services till failover.

► NFS

View the contents of the exports file to ensure that the following file systems are included in rw mode as shown in Example 2-117:

- /home
- /opt/IBM
- /ds_overview

Attention: In general, the software (/home, /ds_overview, and /opt/IBM file systems) is NFS-mounted to the compute nodes from either the front end node or from a NAS device. It is not always from the front end node.

In our case, the front end node uses a SAN for the software and exposes it through NFS. If we had an NAS configuration, there would be no entries required in the exports file for NFS.

Example 2-117 /etc/exports file

```
[root@nile ~]# cat /etc/exports
/home *(rw, sync)
/opt/IBM *(rw, sync)
/ds_overview *(rw, sync)
```

View the status of NFS as shown in Example 2-118.

Example 2-118 NFS status

```
[root@nile ~]# service nfs status
rpc.mountd (pid 5307 5268) is running...
nfsd (pid 5264 5263 5262 5261 5260 5259 5258 5257) is running...
rpc.rquotad (pid 5300 5253) is running...
```

► NIS

The NIS server on the standby node is set up as an NIS subordinate. This was done earlier in setting up the grid infrastructure as described in “NIS” on page 80.

The nisupdatefiles.sh script (shown in Example B-11 on page 263) backs up current local files related to user and hosts (such as /etc/hosts, /etc/shadow, /etc/passwd, and /etc/group), and updates them with information from the NIS server database. This script must be executed on the NIS subordinate server to update the local files (/etc/hosts, /etc/shadow, /etc/passwd, and /etc/group) to sync with the NIS server database information.

Place this script in some location (/root in our case) and run in a cron utility job. To install the above script in the crontab use the following command:

```
# crontab -e
# Run the script to update the local files from NIS server.
# The following will run everyday at 2:15AM.
15 2 * * * /root/nisupdatefiles.sh
```

Note: You must choose an update cycle based on your expected rate of changes to user accounts. The recommended frequency is hourly.

► Virtual IP

Set up the virtual IP the same way as performed on the front end node (as shown in Example 2-119 on page 146 and Example 2-120 on page 146) with only one exception: The ONPARENT=NO must be set on the standby node configuration files for the virtual IP. This option prevents the virtual IP from being activated during a boot sequence. This has to be activated at takeover using the following commands:

```
# ifup eth0:1
# ifup eth1:1
```

Example 2-119 Virtual IP number 1 for eth0

```
[root@nile ~]# cat /etc/sysconfig/network-scripts/ifcfg-eth0:1
DEVICE=eth0:1
BOOTPROTO=static
HWADDR=00:02:55:AC:86:A4
IPADDR=9.43.86.77
NETMASK=255.255.252.0
ONPARENT=no
TYPE=Ethernet
```

Example 2-120 Virtual IP number 1 for eth1

```
[root@nile ~]# cat /etc/sysconfig/network-scripts/ifcfg-eth1:1
DEVICE=eth1:1
BOOTPROTO=static
HWADDR=00:02:55:E4:5B:17
IPADDR=192.168.1.77
NETMASK=255.255.252.0
ONPARENT=no
TYPE=Ethernet
```

Important: The configuration of the additional eth0:1 and eth1:1 files is only required for a manual failover (which is what we did) as opposed to an automated failover. Automated HA solutions perform these activities under the covers and provide virtual IP addresses and running of start/stop scripts based on services associated with a resource group. Resource group is a HA term used to define a set of failover applications, disk mount points, and virtual IP address

2.6 Step 2: Tailoring existing DS/IA jobs to operate in grid environment

The existing IBM Information Server environment has a broad spectrum of DataStage and QualityStage jobs and Information Analyzer jobs. These jobs may be single jobs or job sequences. This allows you to specify a sequence of parallel jobs or server jobs to run. The sequence can also contain control information. For example, you can specify different courses of action to take depending on whether a job in the sequence succeeds or fails. Once you have defined a job sequence, it can be scheduled and run using DataStage Director. It appears in

the DataStage repository and in the DataStage Director client as a job. For details on job sequences, refer to *IBM WebSphere DataStage Designer Client Guide Version 8*, LC18-9893-00

When the IBM Information Server is migrated to a grid environment infrastructure, all the existing jobs also have to be migrated to it. In this section, we describe the steps involved in migrating existing jobs to the grid environment. The migration may involve changes to the parameters only, or changes to both the parameters and stage properties as shown in Figure 2-16 on page 148.

Jobs requiring parameter changes only are described in 2.6.1, “Migrating jobs that require parameter changes only” on page 148.

Jobs requiring parameter and stage property changes are described in 2.6.2, “Migrating jobs requiring parameter and stage property changes” on page 166.

Important: For the grid environment, the environment variables are defined in `grid_global_values` in the `/opt/IBM/InformationServer/Server/PXEngine/grid_enabled.4.0.1/directory`, `DSParams` in the `/opt/IBM/InformationServer/Server/Projects/<project name>` directory, and the Job Run Options. The environment variable precedence is Job Run Options that override `DSParams` (in the `/opt/IBM/InformationServer/Server/Projects/IS_Grid` directory in our case as shown in Example 2-121 on page 147) that overrides `grid_global_values`.

Example 2-121 on page 147 shows the grid toolkit-related environment variables in the `DSParams` file for our `IS_Grid` project in the `/opt/IBM/InformationServer/Server/Projects/IS_Grid` directory. We strongly recommend against modifying the contents of this file.

Example 2-121 DSParams file in /opt/IBM/InformationServer/Server/Projects/IS_Grid directory

```
.....
[EnvVarValues]
"APT_GRID_COMPUTENODES"\1\3"
"APT_GRID_ENABLE"\1\NO"
"APT_GRID_PARTITIONS"\1\3"
```

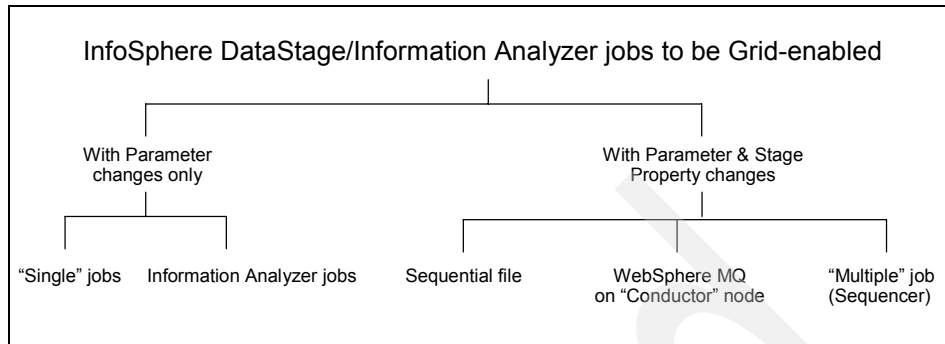


Figure 2-16 Categories of jobs to be migrated to the grid environment

2.6.1 Migrating jobs that require parameter changes only

This section describes examples of a single job and an Information Analyzer job that are migrated to the grid implementation by only adding runtime parameters.

Single jobs

Figure 2-18 on page 150 shows a single job named rowGenSleeper_g that generates the number of rows specified through a parameter, and sleeps for the specified interval (in microseconds) before writing each row out to a Peek stage.

This job requires the following grid environment parameters to be added as shown in Figure 2-17:

- ▶ \$APT_GRID_ENABLE with a prompt of “ENABLE - Use DYNAMIC_CONFIG_SCRIPT (NO/YES/OLD)” and a default string value of NO
- ▶ \$APT_GRID_COMPUTENODES with the prompt “COMPUTENODES - Number of Compute Nodes” and a default string value of 1
- ▶ \$APT_GRID_PARTITIONS with the prompt “PARTITIONS - Number of partitions per Node (compute)” and a default string value of 1

\Jobs\GridEnabled\ParmChangeOnly\rowGenSleeper_g - Job Properties				
General Parameters Job control Dependencies Generated QSH Execution Defaults				
	Parameter name	Prompt	Type	Default Value
1	rows	Number of Rows to Generate	Integer	10
2	useconds	Number of microseconds to sleep	Integer	1000000
3	\$APT_GRID_ENABLE	ENABLE - Use DYNAMIC_CONFIG_SCRIPT (NO/YES/OLD)	String	NO
4	\$APT_GRID_COMPUTENODES	COMPUTENODES - Number of Compute Nodes	String	1
5	\$APT_GRID_PARTITIONS	PARTITIONS - Number of partitions per Node (compute)	String	1

Figure 2-17 Parameters added to Job Properties for rowGenSleeper_g

Perform the following steps to grid-enable the rowGenSleeper_g job:

1. Click the Run icon to run the job in Figure 2-18 on page 150. This brings up the Job Run Options window.
2. Modify the following parameter values in the Job Run Options window as shown in Figure 2-19 on page 151:
 - ENABLE - Use DYNAMIC_CONFIG_SCRIPT (NO/YES/OLD) value to YES
 - COMPUTENODES - Number of Compute Nodes value to 2
 - PARTITIONS - Number of partitions per Node (compute) value to 2

Figure 2-20 on page 151 shows the successful execution of the job where 10 rows are generated and the same number are written to the Peek stage.

Figure 2-21 on page 152 shows the output of the DataStage and QualityStage Director log which shows the servers phoenixpvt.itsosj.sanjose.ibm.com and taruspvt.itsosj.sanjose.ibm.com as the compute nodes, and rowGenSleeper_g_5959.config.apl as the configuration file.

Note: Even though there are no sequential files defined in this job and therefore no variables \$APT_GRID_SEQFILE_HOST and \$APT_GRID_SEQFILE_HOST2 defined in the job, the Director log shows the assignment of the first two nodes (machines) to the \$APT_GRID_SEQFILE_HOST and \$APT_GRID_SEQFILE_HOST2 respectively.

Figure 2-22 on page 152 shows the Event Detail log indicating the parameter values used in the execution of this job.

Figure 2-23 on page 153 shows the Event Detail log indicating the submission of the job to the front end node orion.itsosj.sanjose.ibm.com.

Figure 2-24 on page 153 shows the contents of the generated rowGenSleeper_g_5959.config.apl file which has four nodes in all: Two (Compute1 and Compute2) corresponding to the phoenixpvt.itsosj.sanjose.ibm.com machine, while the other two (Compute3 and Compute4) correspond to the taruspvt.itsosj.sanjose.ibm.com machine. The Conductor node is the kazanpvt.itsosj.sanjose.ibm.com machine.

Example 2-122 on page 154 shows the Director detail output of this execution with key entries highlighted such as the parameter values supplied, the generated configuration file and its contents, as well as the default configuration file contents.

Whenever a job is run in a grid environment, a <jobname>.time file is created within the <JOBDIR> directory. This file contains information about the specific job run. The file contains the following information:

- Start Date (YYYY-MM-DD HH:MM:SS)
- End Date (YYYY-MM-DD HH:MM:SS)
- Seconds (difference between start and end)
- Number of nodes (physical servers)
- Number of partitions (partitions per node)
- User ID used for actual job run
- Queue name job was run within

The output is a single line for each job run. A file can grow up to 200 entries before it begins purging older entries. The contents are separated with a semicolon “;”, including a final semicolon.

Example 2-123 on page 158 shows the contents of the time file for the rowGenSleeper_g job. This entry corresponding to this job is highlighted.

Example 2-124 on page 158 shows the output of the **llmonitor** command while this job is running. It shows one job running. This command erroneously shows it only running on phoenix. Use **llstatus** command for correct information about the two nodes phoenix and tarus being Busy as shown in Example 2-125 on page 159.

Example 2-126 on page 159 shows the output of the **llmonitor** command while after this job has finished. It shows that there is “currently no job status to report”.

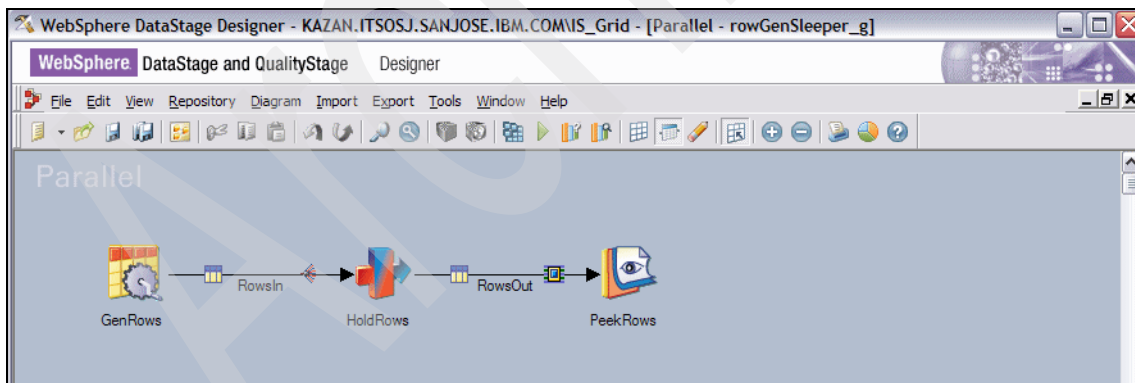


Figure 2-18 Single job rowGenSleeper_g, part 1 of 7

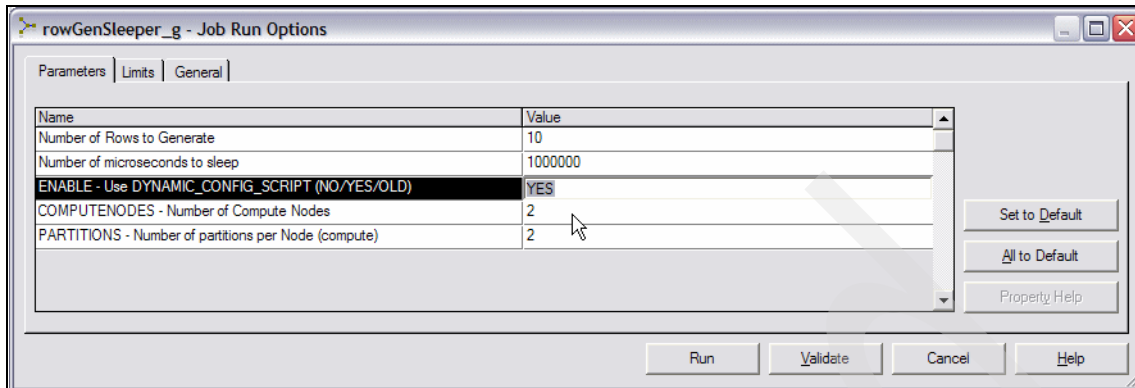


Figure 2-19 Single job rowGenSleeper_g, part 2 of 7

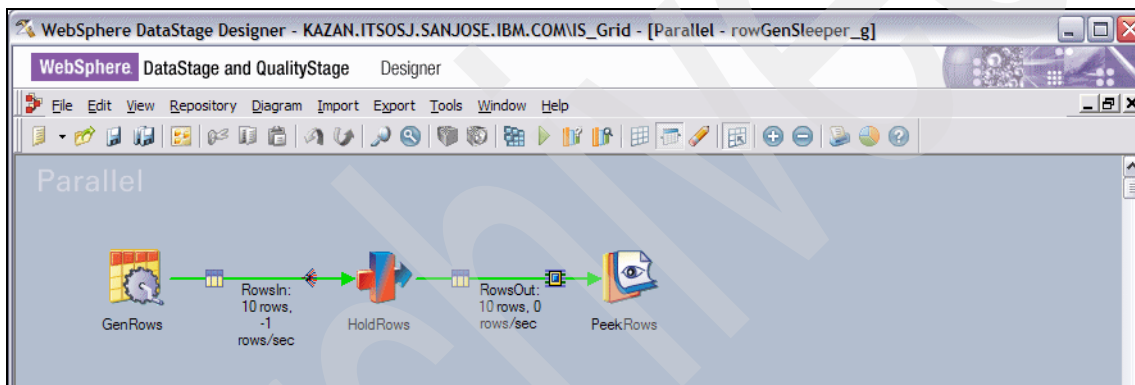


Figure 2-20 Single job rowGenSleeper_g, part 3 of 7

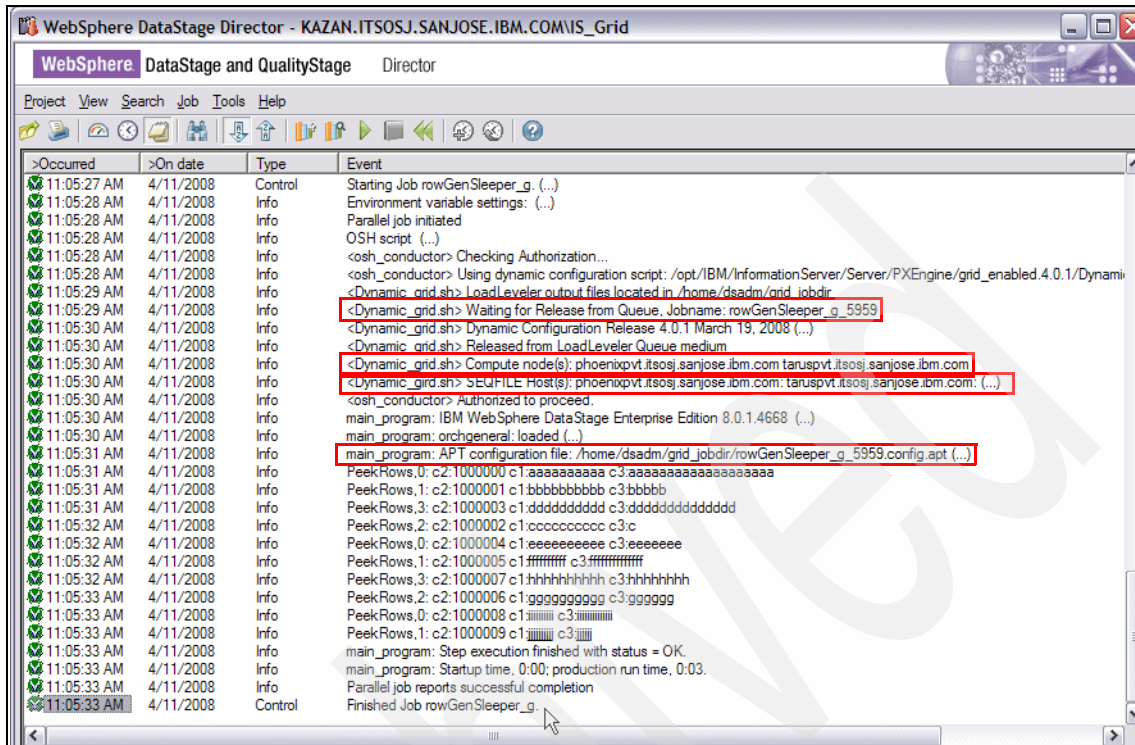


Figure 2-21 Single job rowGenSleeper_g, part 4 of 7

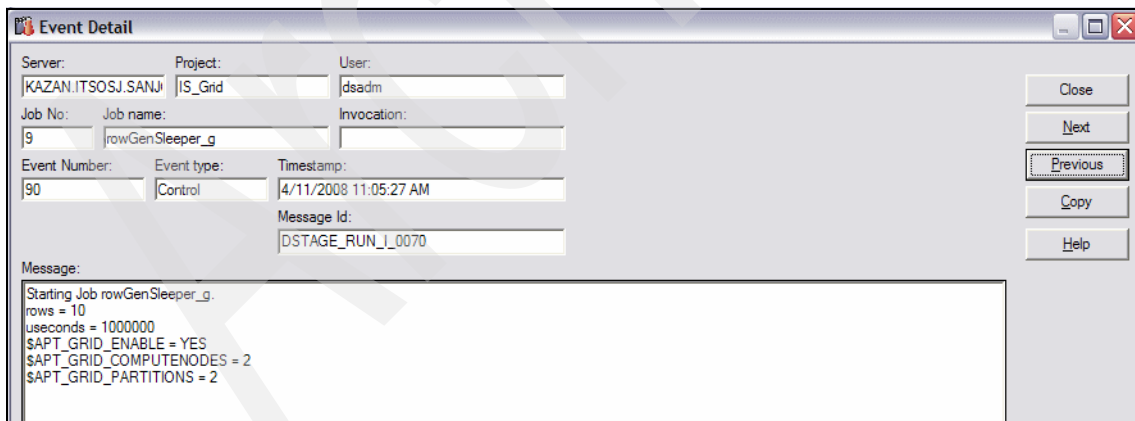


Figure 2-22 Single job rowGenSleeper_g, part 5 of 7

Event Detail		
Server:	Project:	User:
KAZAN.ITSOSJ.SANJi	IS_Grid	dsadm
Job No:	Job name:	Invocation:
9	rowGenSleeper_g	
Event Number:	Event type:	Timestamp:
99	Info	4/11/2008 11:05:30 AM
Message Id:		
		XXXXXP
Message: <Dynamic_grid.sh> Dynamic Configuration Release 4.0.1 March 19, 2008 llsbmit: The job "orion.itsosj.sanjose.ibm.com,37" has been submitted.		

Close
 Next
 Previous
 Copy
 Help

Figure 2-23 Single job rowGenSleeper_g, part 6 of 7

Event Detail		
Server:	Project:	User:
KAZAN.ITSOSJ.SANJi	IS_Grid	dsadm
Job No:	Job name:	Invocation:
9	rowGenSleeper_g	
Event Number:	Event type:	Timestamp:
101	Info	4/11/2008 11:05:30 AM
Message Id:		
		XXXXXS
Message: <pre> <Dynamic_grid.sh> SEQFILE Host(s): phoenixpvt.itsosj.sanjose.ibm.com; taruspvt.itsosj.sanjose.ibm.com; { node "Conductor" { fastname "kazanpvt.itsosj.sanjose.ibm.com" pools "conductor" resource disk "/opt/IBM/InformationServer/Server/Datasets" {pools ""} resource scratchdisk "/tmp" {pools ""} } node "Compute1" { fastname "phoenixpvt.itsosj.sanjose.ibm.com" pools "" resource disk "/opt/IBM/InformationServer/Server/Datasets" {pools ""} resource scratchdisk "/tmp" {pools ""} } node "Compute2" { fastname "phoenixpvt.itsosj.sanjose.ibm.com" pools "" resource disk "/opt/IBM/InformationServer/Server/Datasets" {pools ""} resource scratchdisk "/tmp" {pools ""} } node "Compute3" { fastname "taruspvt.itsosj.sanjose.ibm.com" pools "" resource disk "/opt/IBM/InformationServer/Server/Datasets" {pools ""} resource scratchdisk "/tmp" {pools ""} } node "Compute4" { fastname "taruspvt.itsosj.sanjose.ibm.com" pools "" resource disk "/opt/IBM/InformationServer/Server/Datasets" {pools ""} resource scratchdisk "/tmp" {pools ""} } } </pre>		

Close
 Next
 Previous
 Copy
 Help

Figure 2-24 Single job rowGenSleeper_g, part 7 of 7

Example 2-122 rowGenSleeper_g Director detail log

DataStage Report - Detail Log for job: rowGenSleeper_g
Produced on:4/11/2008 11:03:14 AM
Project: IS_Grid
Host system: KAZAN.ITSOSJ.SANJOSE.IBM.COM
Items: 91 - 120
Sorted on: Date Sorter

Item #: 91
Event ID: 90
Timestamp:: 2008-04-11 11:05:27
Type: Control
User Name: dsadm
Message:: Starting Job rowGenSleeper_g.

rows = 10

useconds = 1000000

\$APT_GRID_ENABLE = YES

\$APT_GRID_COMPUTENODES = 2

\$APT_GRID_PARTITIONS = 2

Item #: 92
Event ID: 91
Timestamp:: 2008-04-11 11:05:28
Type: Info
User Name: dsadm
Message:: Environment variable settings:

APT_COMPILEOPT=-O -fPIC -Wno-deprecated -c

APT_COMPILER=g++

APT_CONFIG_FILE=/opt/IBM/InformationServer/Server/Configurations/default.appt

APT_ERROR_CONFIGURATION=severity, !vseverity, !jobid, moduleid, errorIndex, timestamp, !ipaddr, !nodeplayer, !nodename, opid, message

APT_GRID_COMPUTENODES=2

APT_GRID_CONFIG=

APT_GRID_ENABLE=YES

APT_GRID_IDENTIFIER=

APT_GRID_OPTS=

APT_GRID_PARTITIONS=2

APT_GRID_QUEUE=

APT_GRID_SCRIPTPOST=

APT_GRID_SCRIPTPRE=

APT_GRID_SEQFILE_HOST2=

APT_GRID_SEQFILE_HOST=

APT_GRID_STAT_CMD=


```

APT_LINKER=g++
APT_LINKOPT=-shared -Wl,-Bsymbolic,--allow-shlib-undefined
APT_MONITOR_MINTIME=10
APT_OPERATOR_REGISTRY_PATH=/opt/IBM/InformationServer/Server/Projects/IS_Grid/buildop
APT_ORCHHOME=/opt/IBM/InformationServer/Server/PXEngine
APT_PM_CONDUCTOR_HOSTNAME=kazanpvt.itsosj.sanjose.ibm.com
ASBHOME=/opt/IBM/InformationServer/ASBNode
BELL=^G
CAC_CONFIG=/opt/ibm/wsclassic91/cli/lib/cac.ini
CLASSPATH=./opt/IBM/InformationServer/Server/DSEngine/java/lib
DB2DIR=/opt/IBM/DB2
DB2INSTANCE=db2inst1
DSHOME=/opt/IBM/InformationServer/Server/DSEngine
DSIPC_OPEN_TIMEOUT=30
DS_ENABLE_RESERVED_CHAR_CONVERT=0
DS_HOSTNAME_ALIAS=kazan.itsosj.sanjose.ibm.com
DS_OPERATOR_BUILDOP_DIR=buildop
DS_OPERATOR_WRAPPED_DIR=wrapped
DS_TDM_PIPE_OPEN_TIMEOUT=720
DS_TDM_TRACE_SUBROUTINE_CALLS=0
DS_USERNO=-5914
FLAVOR=-1
GRIDHOME=/opt/IBM/InformationServer/Server/PXEngine/grid_enabled.4.0.1
HOME=/home/dsadm
INSTHOME=/home/db2inst1
JAVAPATH=/opt/IBM/InformationServer/ASBNode/apps/jre/bin
LANG=en_US.UTF-8
LD_LIBRARY_PATH=/opt/IBM/InformationServer/Server/Projects/IS_Grid/buildop:/opt/IBM/InformationServer/Server/DSComponents/lib:/opt/IBM/InformationServer/Server/DSComponents/bin:/opt/IBM/InformationServer/Server/DSParallel:/opt/IBM/InformationServer/Server/PXEngine/user_lib:/opt/IBM/InformationServer/Server/PXEngine/lib:/opt/IBM/InformationServer/Server/Projects/IS_Grid/RT_BP9.0:/opt/IBM/InformationServer/_jvm/jre/bin/classic:/opt/ibm/wsclassic91/cli/lib:/opt/IBM/InformationServer/Server/brand_
ed_odbc/lib:/opt/IBM/InformationServer/Server/DSEngine/lib:/opt/IBM/InformationServer/Server/DSEngine/uvdls:/opt/IBM/InformationServer/ASBNode/apps/jre/bin:/opt/IBM/InformationServer/ASBNode/apps/jre/bin/classic:/opt/IBM/InformationServer/ASBNode/lib/cpp:/opt/IBM/InformationServer/ASBNode/apps/proxy/cpp/linux-all-x86:/home/db2inst1/sqlib/lib:/opt/IBM/db2/V9/lib32:/opt/mqm:/var/mqm:/opt/mqm/lib:/opt/mqm/bin:/lib:/usr/lib:
.
LOGNAME=dsadm
ODBCINI=/opt/IBM/InformationServer/Server/DSEngine/.odbc.ini
OLDPWD=/

```

```

OSH_STDOUT_MSG=1
PATH=/opt/IBM/InformationServer/Server/Projects/IS_Grid/wrapped:/opt/IBM/InformationServer/Server/Projects/IS_Grid/buildop:/opt/IBM/InformationServer/Server/Projects/IS_Grid/RT_BP9.0:/opt/IBM/InformationServer/Server/DSComponents/lib:/opt/IBM/InformationServer/Server/DSComponents/bin:/opt/IBM/InformationServer/Server/DSParallel:/opt/IBM/InformationServer/Server/PXEngine/user_osh_wrappers:/opt/IBM/InformationServer/Server/PXEngine/osh_wrappers:/opt/IBM/InformationServer/Server/PXEngine/bin:/opt/IBM/InformationServer/Server/PXEngine/grid_enabled.4.0.1:/opt/IBM/InformationServer/ASBNode/apps/jre/bin:/opt/IBM/InformationServer/_jvm/jre/bin:/sbin:/usr/sbin:/home/db2inst1/sqllib/bin:/home/db2inst1/sqllib/adm:/home/db2inst1/sqllib/misc:/home/loadl/bin:/usr/kerberos/bin:/usr/local/bin:/bin:/usr/bin:/usr/X11R6/bin:.
PIDTOK=2297
PWD=/opt/IBM/InformationServer/Server/DSEngine
PX_DBCONNECTHOME=/opt/IBM/InformationServer/Server/DSComponents
SHELL=/bin/bash
SHLVL=4
TERM=
THREADS_FLAG=native
UDTBIN=/opt/IBM/InformationServer/Server/DSEngine/ud41/bin
UDTHOME=/opt/IBM/InformationServer/Server/DSEngine/ud41
USER=dsadm
WHO=IS_Grid
_=/usr/bin/nohup
.....
  Message:: <Dynamic_gird.sh> SEQFILE Host(s):
phoenixpvt.itsosj.sanjose.ibm.com: taruspvt.itsosj.sanjose.ibm.com:
{
  node "Conductor"
  {
    fastname "kazanpvt.itsosj.sanjose.ibm.com"
    pools "conductor"
    resource disk "/opt/IBM/InformationServer/Server/Datasets" {pools ""}
    resource scratchdisk "/tmp" {pools ""}
  }
  node "Compute1"
  {
    fastname "phoenixpvt.itsosj.sanjose.ibm.com"
    pools ""
    resource disk "/opt/IBM/InformationServer/Server/Datasets" {pools ""}
    resource scratchdisk "/tmp" {pools ""}
  }
  node "Compute2"
  {

```

```

    fastname "phoenixpvt.itsosj.sanjose.ibm.com"
    pools ""
    resource disk "/opt/IBM/InformationServer/Server/Datasets" {pools ""}
    resource scratchdisk "/tmp" {pools ""}
}
node "Compute3"
{
    fastname "taruspvt.itsosj.sanjose.ibm.com"
    pools ""
    resource disk "/opt/IBM/InformationServer/Server/Datasets" {pools ""}
    resource scratchdisk "/tmp" {pools ""}
}
node "Compute4"
{
    fastname "taruspvt.itsosj.sanjose.ibm.com"
    pools ""
    resource disk "/opt/IBM/InformationServer/Server/Datasets" {pools ""}
    resource scratchdisk "/tmp" {pools ""}
}
}

```

.....

Message:: main_program: APT configuration file:
/home/dsadm/grid_jobdir/rowGenSleeper_g_5959.config.ap

```

{
    node "node1"
    {
        fastname "kazanpvt.itsosj.sanjose.ibm.com"
        pools "conductor"
        resource disk "/opt/IBM/InformationServer/Server/Datasets" {pools
""}
        resource scratchdisk "/opt/IBM/InformationServer/Server/Scratch"
{pools ""}
    }
    node "node2"
    {
        fastname "kazanpvt.itsosj.sanjose.ibm.com"
        pools ""
        resource disk "/opt/IBM/InformationServer/Server/Datasets" {pools
""}
        resource scratchdisk "/opt/IBM/InformationServer/Server/Scratch"
{pools ""}
    }
    node "node3"
    {
        fastname "kazanpvt.itsosj.sanjose.ibm.com"

```

```

        pools ""
        resource disk "/opt/IBM/InformationServer/Server/Datasets" {pools
""}
        resource scratchdisk "/opt/IBM/InformationServer/Server/Scratch"
{pools ""}
    }
}
.....

```

End of report.

Example 2-123 Time file in the /home/dsadm/grid_jobdir/ directory

```

2008-04-11 11:05:27;2008-04-11 11:05:33;06;2;2;dsadm;No_Class;
2008-04-14 15:45:45;2008-04-14 15:46:18;33;1;3;dsadm;No_Class;
2008-04-16 09:49:02;2008-04-16 09:49:35;33;3;5;dsadm;No_Class;
2008-04-18 14:25:23;2008-04-18 14:26:58;95;3;2;dsadm;No_Class;
2008-04-18 14:35:48;2008-04-18 14:36:21;33;2;2;dsadm;No_Class;

```

Example 2-124 llmonitor command output while job is running

Fri Apr 11 11:06:02 PDT 2008

Name	Schedd	InQ	Act	Startd	Run	LdAvg	Idle	Arch	OpSys
luxor.itsosj.sanjose.ibm.	Avail	0	0	Idle	0	1.00	9999	i386	Linux2
orion.itsosj.sanjose.ibm.	Avail	1	1	None	0	0.23	52	i386	Linux2
phoenix.itsosj.sanjose.ib	Avail	0	0	Busy	1	0.00	9999	i386	Linux2
tarus.itsosj.sanjose.ibm.	Avail	0	0	Busy	1	0.00	9999	i386	Linux2

i386/Linux2	4 machines	1 jobs	2 running
Total Machines	4 machines	1 jobs	2 running

The Central Manager is defined on orion.itsosj.sanjose.ibm.com

The BACKFILL scheduler is in use

All machines on the machine_list are present.

Id	Owner	Submitted	ST	PRI	Class	Running On
orion.37.0	dsadm	4/11 11:05	R	50	medium	phoenix

1 job step(s) in queue, 0 waiting, 0 pending, **1 running**, 0 held, 0 preempted

Example 2-125 llstatus output while job is running

```
....
Name Schedd InQ Act Startd Run LdAvg Idle Arch OpSys
luxor.itsosj.sanjose.ibm. Avail 0 0 Idle 0 1.00 9999 i386 Linux2
orion.itsosj.sanjose.ibm. Avail 1 1 None 0 0.23 52 i386 Linux2
phoenix.itsosj.sanjose.ib Avail 0 0 Busy 1 0.00 9999 i386 Linux2
tarus.itsosj.sanjose.ibm. Avail 0 0 Busy 1 0.00 9999 i386 Linux2
....
```

Example 2-126 llmonitor command output after job has finished

Fri Apr 11 11:06:05 PDT 2008

Name	Schedd	InQ	Act	Startd	Run	LdAvg	Idle	Arch	OpSys
luxor.itsosj.sanjose.ibm.	Avail	0	0	Idle	0	1.00	9999	i386	Linux2
orion.itsosj.sanjose.ibm.	Avail	1	1	None	0	0.23	52	i386	Linux2
phoenix.itsosj.sanjose.ib	Avail	0	0	Idle	0	0.14	9999	i386	Linux2
tarus.itsosj.sanjose.ibm.	Avail	0	0	Idle	0	0.00	9999	i386	Linux2
i386/Linux2	4 machines			1 jobs		0	running		
Total Machines	4 machines			1 jobs		0	running		

The Central Manager is defined on orion.itsosj.sanjose.ibm.com

The BACKFILL scheduler is in use

All machines on the machine_list are present.

llq: There is currently no job status to report.

Information Analyzer jobs

To enable Information Analyzer jobs, you must modify the project defaults as follows:

- ▶ \$APT_GRID_ENABLE = YES
- ▶ \$APT_GRID_PARTITIONS = 6 to maximize parallelism on each compute node

We had an existing Information Analyzer job that was created during the residency for the Redbooks publication *IBM WebSphere Information Analyzer and Data Quality Assessment*, SG24-7508. That job (BaseProfile_Column_Analysis_Task21780134) was created in the ANALYZERPROJECT and performed a column analysis of the IA.EMPLOYEE table (Figure 2-25 on page 160) in the IASAMPLE database. The Director log output of this job is shown in Figure 2-26 on page 161. For details on generating,

running and monitoring Information Analyzer jobs, refer to the Redbooks publication *IBM WebSphere Information Analyzer and Data Quality Assessment*, SG24-7508.

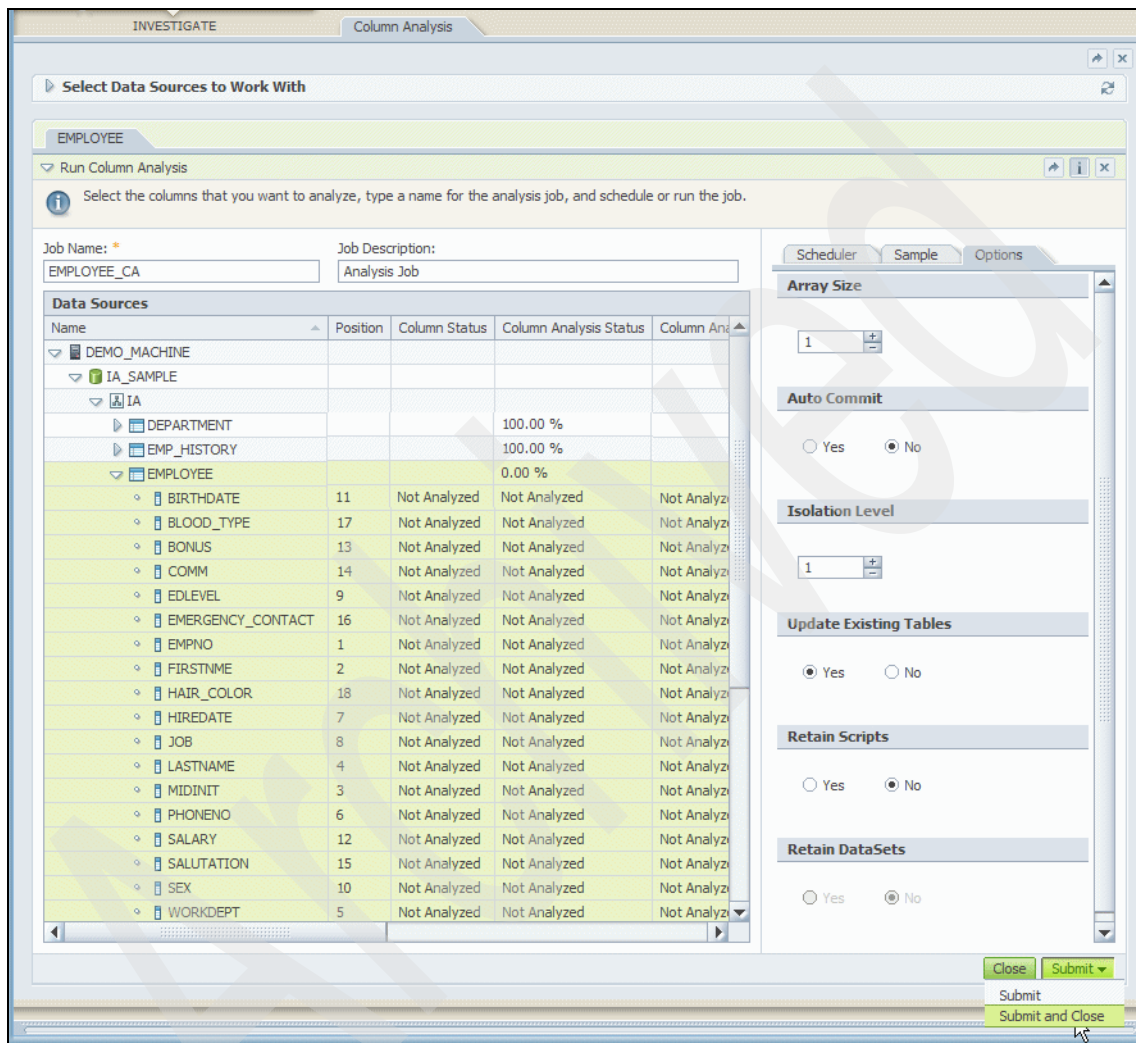


Figure 2-25 Employee table on which column analysis was performed

>Occurred	>On date	Type	Event
10:36:25 AM	8/6/2007	Control	Starting Job BaseProfile_Column_Analysis_Task21780134.
10:36:25 AM	8/6/2007	Info	Environment variable settings: (...)
10:36:25 AM	8/6/2007	Info	Parallel job initiated
10:36:26 AM	8/6/2007	Info	main_program: IBM WebSphere DataStage Enterprise Edition 8.0.1.4458 (...)
10:36:26 AM	8/6/2007	Info	main_program: orchgeneral: loaded (...)
10:36:26 AM	8/6/2007	Info	pxbridge: No design time SQL datatype provided for field LASTNAME (...)
10:36:26 AM	8/6/2007	Info	pxbridge: [IIS-CONN-ODBC-000105] Connected to UDB DB2 for Windows, U...
10:36:27 AM	8/6/2007	Info	main_program: APT configuration file: /opt/IBM/InformationServer/Server/Co...
10:36:27 AM	8/6/2007	Warning	generator: When checking operator: When binding output schema variable "r...
10:36:27 AM	8/6/2007	Warning	generator: When checking operator: When binding output schema variable "r...
10:36:27 AM	8/6/2007	Warning	generator: When checking operator: When binding output schema variable "r...
10:36:27 AM	8/6/2007	Warning	generator: When checking operator: When binding output schema variable "r...
10:36:30 AM	8/6/2007	Info	generator: When checking operator: Field DomainValueFlagDate of type date...
10:36:32 AM	8/6/2007	Info	pxbridge,0: [IIS-CONN-ODBC-000105] Connected to UDB DB2 for Windows, ...
10:36:32 AM	8/6/2007	Info	main_program: Step execution finished with status = OK.
10:36:32 AM	8/6/2007	Info	main_program: Startup time, 0:05; production run time, 0:00.
10:36:32 AM	8/6/2007	Info	Parallel job reports successful completion
10:36:32 AM	8/6/2007	Control	Finished Job BaseProfile_Column_Analysis_Task21780134.

Log for job: BaseProfile_Column_Analysis_Task21780134 18 entries (filtered) Server time: 8/6/2007 10:37 AM

Figure 2-26 Director log output of job BaseProfile_Column_Analysis_Task21780134 (pre-grid)

Perform the following steps to grid-enable the BaseProfile_Column_Analysis_Task21780134 job:

1. Launch DataStage and QualityStage Administrator Client and navigate to the Project Properties - KAZAN.ITSOSJ.SANJOSE.IBM.COM\ANALYZERPROJECT window and click **Environment...** as shown in Figure 2-27 on page 163.
2. Select **Grid** in the navigation tree to highlight all the grid environment variables and set the \$APT_GRID_COMPUTENODES variable to 1 as shown in Figure 2-28 on page 163. Click **OK**.

Note: The APT_GRID_COMPUTENODES value must not be greater than 1. The job created by Information Analyzer uses the ODBC common connector to perform all reads.

Having a value greater than 1 creates bottlenecks in the process flow because the input from the source is always sequential. This means that the resources on multiple compute nodes are never fully used, regardless of the number of partitions used.

Setting the value to one and partitions to six allows the single compute node to create more processes and process the evaluation of data after it is sourced, but it still might not use all of the resources on the first compute node, even with the partitions set to 6.

The only reason to increase APT_GRID_COMPUTENODES to greater than one is if the scratch space available on a single node does not allow for the sorts required during the evaluation. This is only an issue if the data being sourced is greater than the scratch disk space available.

3. Set the \$APT_GRID_ENABLE variable to YES.
4. Set the \$APT_GRID_PARTITIONS variable to 6.
5. Run the job BaseProfile_Column_Analysis_Task21780134 (not shown here) and view its execution in the Director log as shown in Figure 2-29 on page 164. It shows the following information:
 - One compute node selected as being phoenixpvt.itsosj.sanjose.ibm.com
 - SEQFILE Host(s) being phoenixpvt.itsosj.sanjose.ibm.com and phoenixpvt.itsosj.sanjose.ibm.com

Note: Information Analyzer never uses flat files. Therefore, the contents of APT_GRID_SEQFILE_HOST and APT_GRID_SEQFILE_HOST2 are never used.

- The name of the generated configuration file
BaseProfile_Column_Analysis_Task21780134_18850.config.apl

Figure 2-30 on page 165 and Figure 2-31 on page 166 show the contents of the generated configuration file showing a total of six compute nodes (Compute1 through Compute6) processing this job.

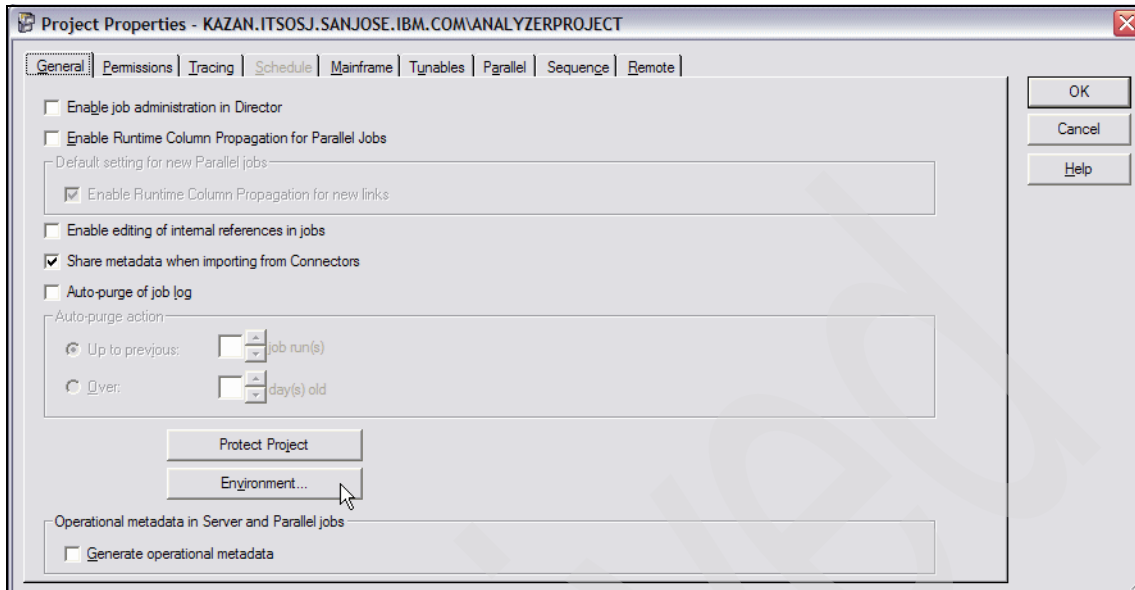


Figure 2-27 Information Analyzer job, part 1 of 5

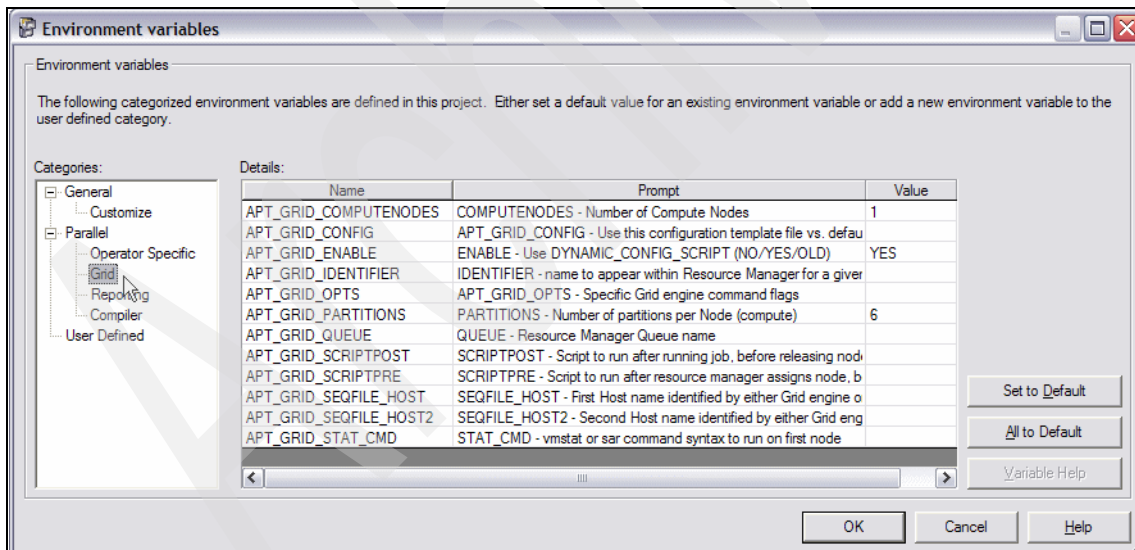


Figure 2-28 Information Analyzer job, part 2 of 5

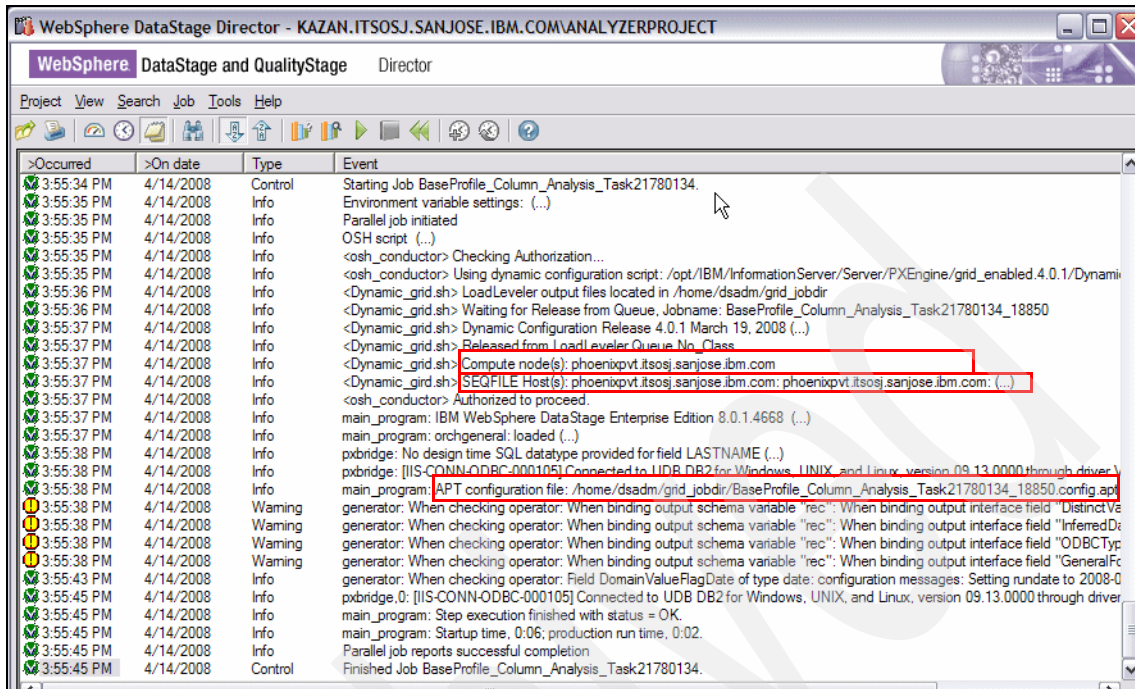


Figure 2-29 Information Analyzer job, part 3 of 5

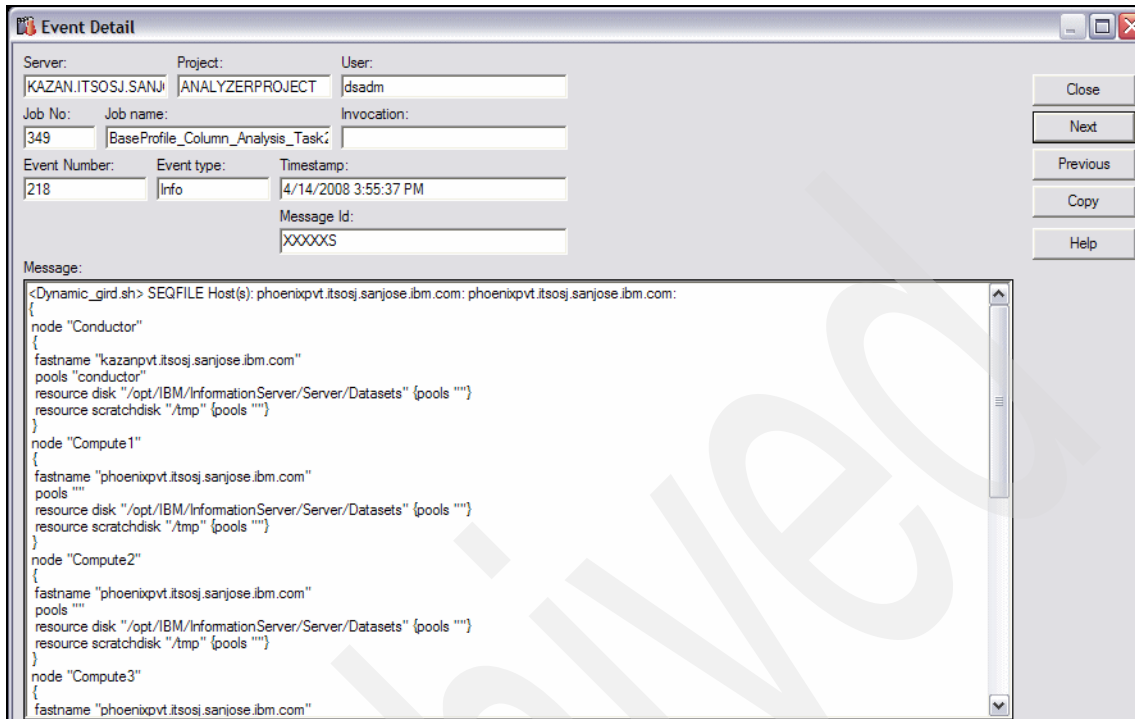


Figure 2-30 Information Analyzer job, part 4 of 5

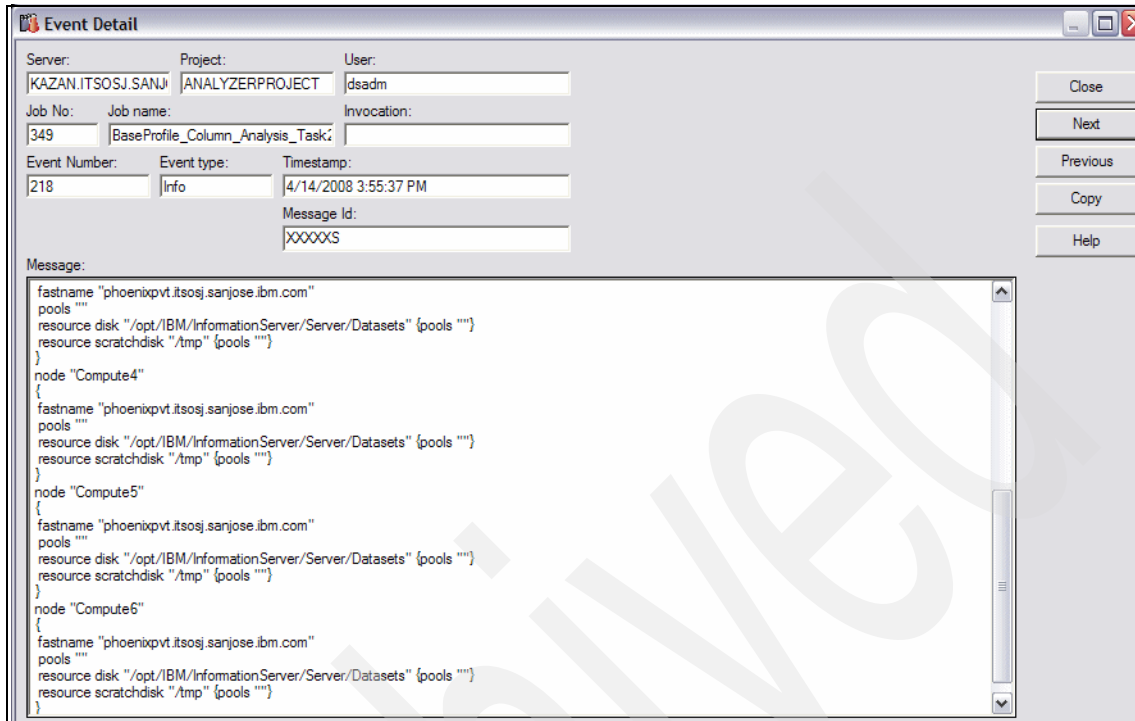


Figure 2-31 Information Analyzer job, part 5 of 5

2.6.2 Migrating jobs requiring parameter and stage property changes

Some existing jobs not only require parameter changes, but also the addition and modification of stage properties with the job.

Three such examples are shown here as follows:

- ▶ Single job with a sequential file
- ▶ Single job with a WebSphere MQ Connector stage
- ▶ Job sequence

Permutation combinations of these examples occur in real world environments and you must be able to extrapolate these examples to them.

Single job with a sequential file

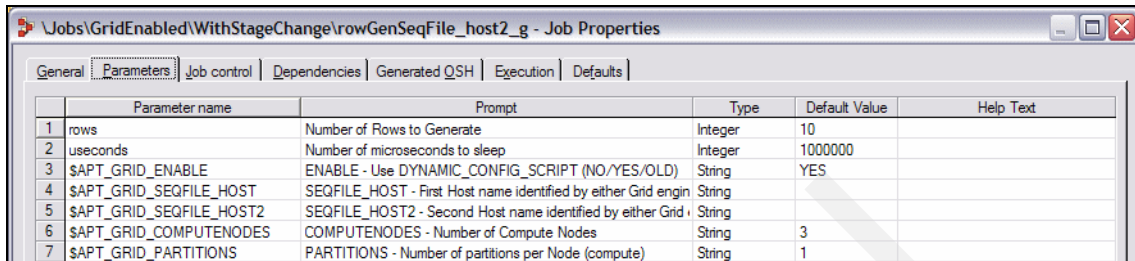
In order to read or write sequential files on a grid, all path name entries must be modified to include a prefix which identifies the host responsible for read or writing the actual file. Because you do not know the actual host name which is assigned, you must define a variable that is set by the toolkit.

Two variables (\$APT_GRID_SEQFILE_HOST and \$APT_GRID_SEQFILE_HOST2) are updated to reflect the first two hosts assigned by the resource manager. When run on a job-by-job basis, the toolkit modifies the contents of these parameters based on the values assigned by the resource manager. Each includes an actual host name followed by a colon, such as "phoenixpvt:". If there is only one node assigned, both variables have the same value.

A DataStage job with a sequential file stage that must be grid-enabled must be modified as follows:

1. Add the following grid environment parameters as shown in Figure 2-32 on page 168:
 - \$APT_GRID_ENABLE with a prompt of ENABLE - Use DYNAMIC_CONFIG_SCRIPT (NO/YES/OLD) and a default string value of NO
 - \$APT_GRID_SEQFILE_HOST with a prompt SEQFILE_HOST - First Host name identified by either Grid Engine or from IONODE names (job parameter returned) with a default of blank
 - \$APT_GRID_SEQFILE_HOST2 with a prompt SEQFILE_HOST - Second Host name identified by either Grid Engine or from IONODE names (job parameter returned) with a default of blank
 - \$APT_GRID_COMPUTENODES with the prompt COMPUTENODES - Number of Compute Nodes and a default string value of 1
 - \$APT_GRID_PARTITIONS with the prompt PARTITIONS - Number of partitions per Node (compute) and a default string value of 1

Note: Even though our job rowGenSeqFile_host2_g had only one sequential file and therefore only the \$APT_GRID_SEQFILE_HOST parameter sufficed, we added \$APT_GRID_SEQFILE_HOST2 as good practice.



	Parameter name	Prompt	Type	Default Value	Help Text
1	rows	Number of Rows to Generate	Integer	10	
2	useconds	Number of microseconds to sleep	Integer	1000000	
3	\$APT_GRID_ENABLE	ENABLE - Use DYNAMIC_CONFIG_SCRIPT (NO/YES/OLD)	String	YES	
4	\$APT_GRID_SEQFILE_HOST	SEQFILE_HOST - First Host name identified by either Grid engine	String		
5	\$APT_GRID_SEQFILE_HOST2	SEQFILE_HOST2 - Second Host name identified by either Grid	String		
6	\$APT_GRID_COMPUTENODES	COMPUTENODES - Number of Compute Nodes	String	3	
7	\$APT_GRID_PARTITIONS	PARTITIONS - Number of partitions per Node (compute)	String	1	

Figure 2-32 Parameters added to Job Properties for rowGenSeqFile_host2_g

2. Modify the File = attribute under the Property tab in the Sequential File stage tab to include '#\$APT_GRID_SEQFILE_HOST#' preceding the filename as shown in Figure 2-33.

If there are multiple files, you may choose to include
 # \$APT_GRID_SEQFILE_HOST2# if you want to have that file accessed from
 a node different from the one assigned to the node you associated with
 # \$APT_GRID_SEQFILE_HOST#.

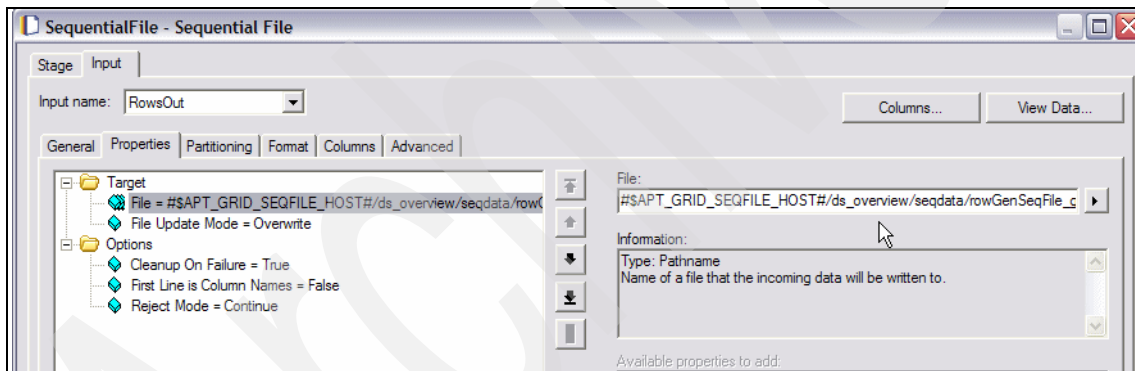


Figure 2-33 \$APT_GRID_SEQFILE_HOST environment variable in Sequential file stage

Note: The toolkit substitutes the first compute node returned from the resource manager into the \$APT_GRID_SEQFILE_HOST environment variable, and the second compute node returned from the resource manager into the \$APT_GRID_SEQFILE_HOST2 environment variable. When the resource manager returns only one compute node, the same value is substituted into both environment variables.

Perform the following steps to grid-enable the rowGenSeqFile_host2_g job:

1. Figure 2-34 on page 170 shows a single job named rowGenSeqFile_host2_g that generates the number of rows specified through a parameter, and sleeps for the specified interval (in microseconds) before writing each row out to a sequential file.
2. Modify the File attribute by prepending the file name with the string `#$APT_GRID_SEQFILE_HOST#` in the Target category under the Properties tab in the Input page of the Sequential File stage. See Figure 2-33 on page 168.
3. Click the Run icon to run the job in Figure 2-34 on page 170 which brings up the Job Run Options window.
4. Modify the following parameter values in the Job Run Options window as shown in Figure 2-35 on page 171:
 - ENABLE - Use DYNAMIC_CONFIG_SCRIPT (NO/YES/OLD) value to YES
 - \$APT_GRID_SEQFILE_HOST with a prompt SEQFILE_HOST - First Host name identified by either Grid Engine or from IONODE names (job parameter returned) to blank
 - \$APT_GRID_SEQFILE_HOST2 with a prompt SEQFILE_HOST - Second Host name identified by either Grid Engine or from IONODE names (job parameter returned) to blank
 - COMPUTENODES - Number of Compute Nodes value to 3
 - PARTITIONS - Number of partitions per Node (compute) value to 1

Figure 2-36 on page 171 shows the successful execution of the job where 10 rows are generated and the same number are written to the sequential file.

Figure 2-37 on page 172 shows the output of the DataStage and QualityStage Director log which shows the servers luxorpvt.itsosj.sanjose.ibm.com, phoenixpvt.itsosj.sanjose.ibm.com, and taruspvt.itsosj.sanjose.ibm.com as the compute nodes, and rowGenSeqFile_host2_g_26895.config.apl as the configuration file.

Note: Even though there is only one sequential file defined in this job and there are two variables (\$APT_GRID_SEQFILE_HOST and \$APT_GRID_SEQFILE_HOST2) defined in the job, the Director log shows the assignment of the first two nodes (machines) (luxorpvt.itsosj.sanjose.ibm.com and phoenixpvt.itsosj.sanjose.ibm.com) to the \$APT_GRID_SEQFILE_HOST and \$APT_GRID_SEQFILE_HOST2 respectively.

The Event Detail log (Figure 2-38 on page 172) indicates the parameter values used in the execution of this job.

The Event Detail log (Figure 2-39 on page 173) indicates the compute nodes used in the execution of this job.

Figure 2-40 on page 173 shows the SEQFILE Host(s) luxorpvt.itsosj.sanjose.ibm.com and phoenixpvt.itsosj.sanjose.ibm.com and the contents of the generated rowGenSeqFile_host2_g_26895.config.apf file which has three nodes in all, one each corresponding to the following machines:

- luxorpvt.itsosj.sanjose.ibm.com
- phoenixpvt.itsosj.sanjose.ibm.com
- taruspvt.itsosj.sanjose.ibm.com

The Conductor node is the kazanpvt.itsosj.sanjose.ibm.com machine.

Example 2-127 on page 174 shows the rowGenSeqFile_host2_g job Director detail output of this execution with key entries highlighted such as the parameter values supplied, the generated configuration file and its contents, as well as the default configuration file contents. The time files and **llmonitor** command output is not shown here.

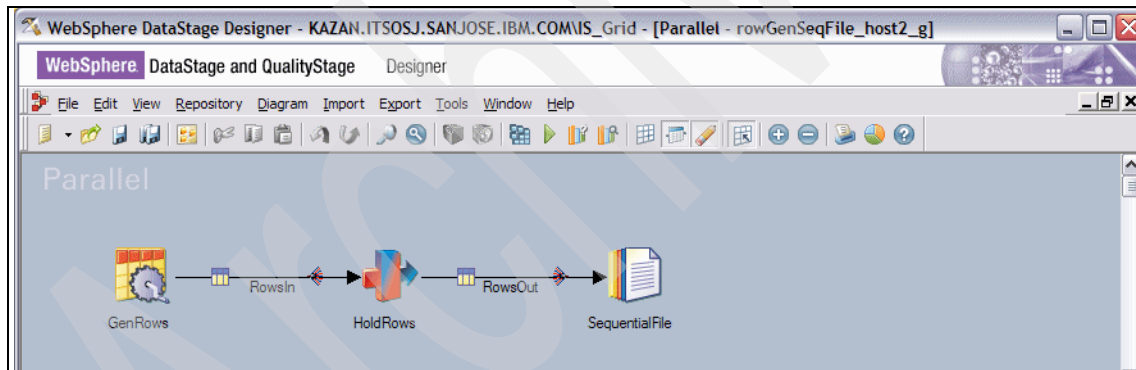


Figure 2-34 Sequential file, part 1 of 7

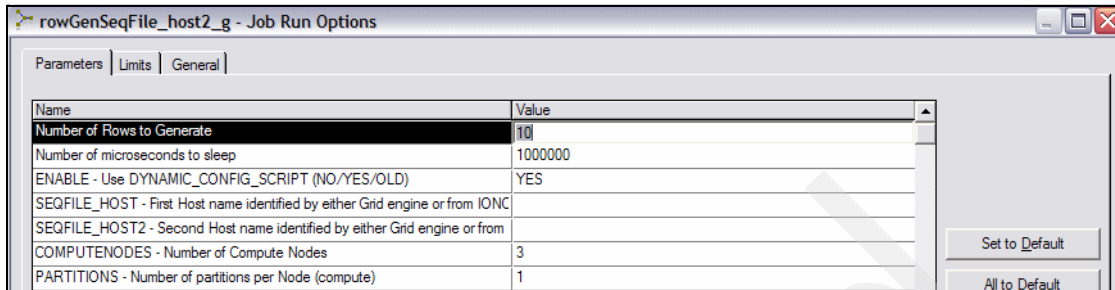


Figure 2-35 Sequential file, part 2 of 7

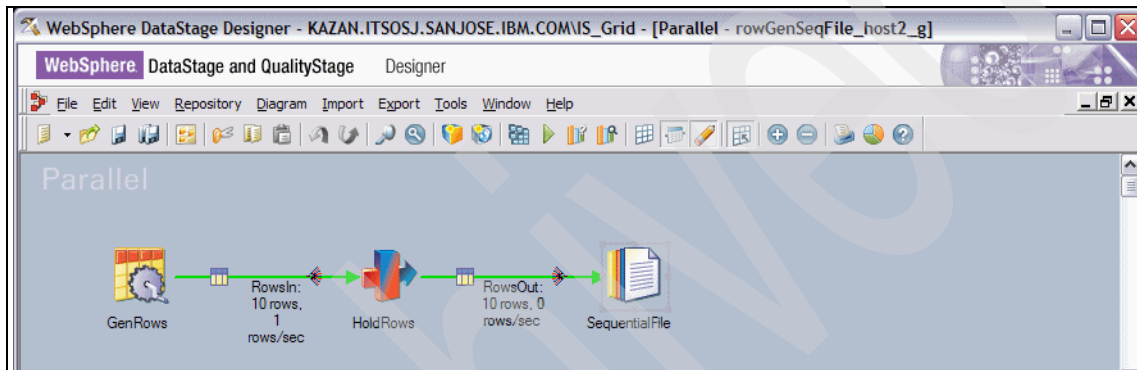


Figure 2-36 Sequential file, part 3 of 7

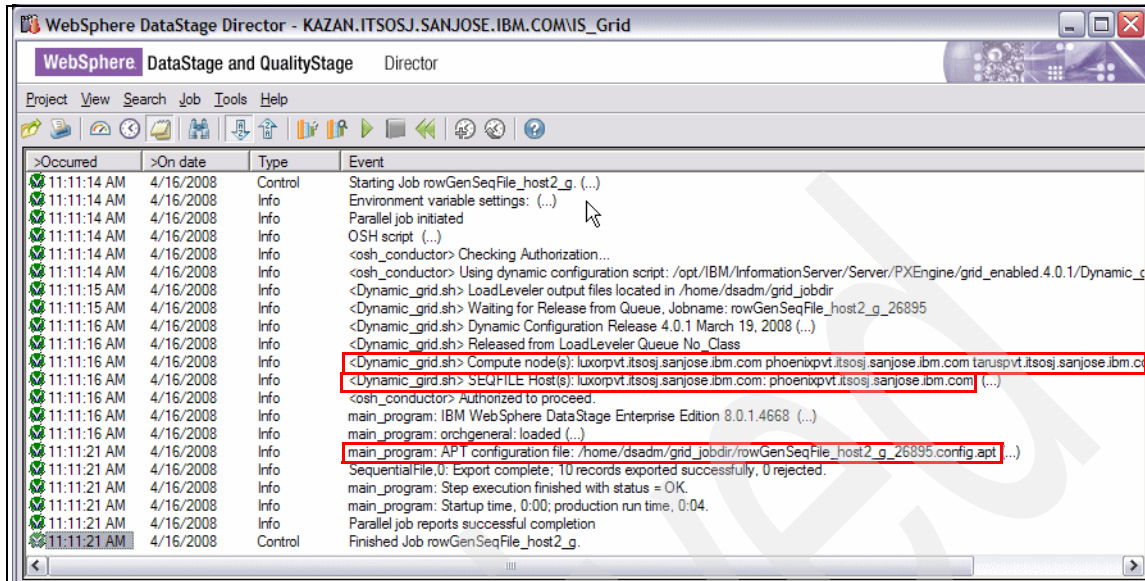


Figure 2-37 Sequential file, part 4 of 7

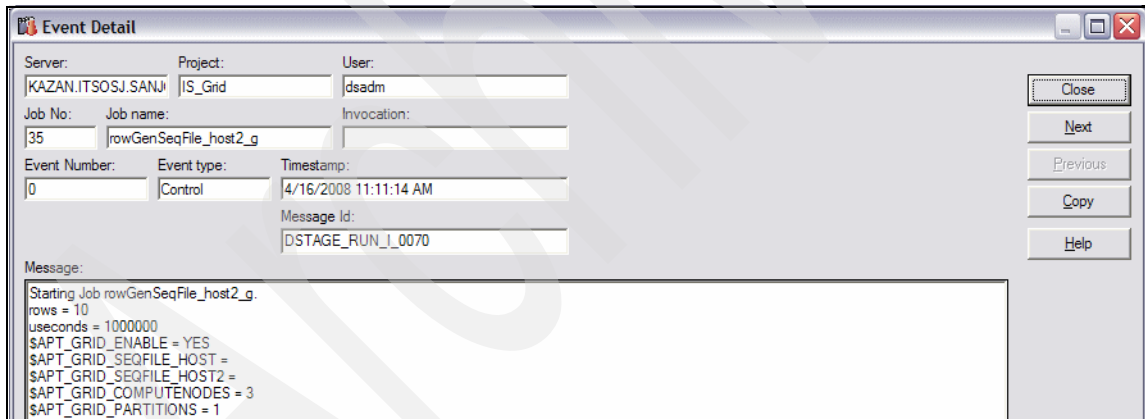


Figure 2-38 Sequential file, part 5 of 7

Event Detail			
Server:	Project:	User:	
KAZAN.ITSOSJ.SANJi	IS_Grid	dsadm	
Job No:	Job name:	Invocation:	
35	rowGenSeqFile_host2_g		
Event Number:	Event type:	Timestamp:	
10	Info	4/16/2008 11:11:16 AM	
		Message Id:	
		XXXXXR	
Message:			
<Dynamic_grid.sh> Compute node(s): luxorpvt.itsosj.sanjose.ibm.com phoenixpvt.itsosj.sanjose.ibm.com taruspvt.itsosj.sanjose.ibm.com			

Close
Next
Previous
Copy
Help

Figure 2-39 Sequential file, part six of 7

Event Detail			
Server:	Project:	User:	
KAZAN.ITSOSJ.SANJi	IS_Grid	dsadm	
Job No:	Job name:	Invocation:	
35	rowGenSeqFile_host2_g		
Event Number:	Event type:	Timestamp:	
11	Info	4/16/2008 11:11:16 AM	
		Message Id:	
		XXXXXS	
Message:			
<pre><Dynamic_grid.sh> SEQFILE Host(s): luxorpvt.itsosj.sanjose.ibm.com: phoenixpvt.itsosj.sanjose.ibm.com: { node "Conductor" { fastname "kazanpvt.itsosj.sanjose.ibm.com" pools "" resource disk "/opt/IBM/InformationServer/Server/Datasets" {pools ""} resource scratchdisk "/tmp" {pools ""} } node "Compute1" { fastname "luxorpvt.itsosj.sanjose.ibm.com" pools "" resource disk "/opt/IBM/InformationServer/Server/Datasets" {pools ""} resource scratchdisk "/tmp" {pools ""} } node "Compute2" { fastname "phoenixpvt.itsosj.sanjose.ibm.com" pools "" resource disk "/opt/IBM/InformationServer/Server/Datasets" {pools ""} resource scratchdisk "/tmp" {pools ""} } node "Compute3" { fastname "taruspvt.itsosj.sanjose.ibm.com" pools "" resource disk "/opt/IBM/InformationServer/Server/Datasets" {pools ""} resource scratchdisk "/tmp" {pools ""} } }</pre>			

Close
Next
Previous
Copy
Help

Figure 2-40 Sequential file, part 7 of 7

DataStage Report - Detail Log for job: rowGenSeqFile_host2_g

Produced on: 4/16/2008 11:09:52 AM

Project: IS_Grid

Host system: KAZAN.ITSOSJ.SANJOSE.IBM.COM

.....

rows = 10

useconds = 1000000

\$APT_GRID_ENABLE = YES

\$APT_GRID_SEQFILE_HOST =

\$APT_GRID_SEQFILE_HOST2 =

\$APT_GRID_COMPUTENODES = 3

\$APT_GRID_PARTITIONS = 1

.....

Message:: <Dynamic_gird.sh> **SEQFILE Host(s):**

luxorpvt.itsosj.sanjose.ibm.com: phoenixpvt.itsosj.sanjose.ibm.com:

```
{
  node "Conductor"
  {
    fastname "kazanpvt.itsosj.sanjose.ibm.com"
    pools "conductor"
    resource disk "/opt/IBM/InformationServer/Server/Datasets" {pools ""}
    resource scratchdisk "/tmp" {pools ""}
  }
  node "Compute1"
  {
    fastname "luxorpvt.itsosj.sanjose.ibm.com"
    pools ""
    resource disk "/opt/IBM/InformationServer/Server/Datasets" {pools ""}
    resource scratchdisk "/tmp" {pools ""}
  }
  node "Compute2"
  {
    fastname "phoenixpvt.itsosj.sanjose.ibm.com"
    pools ""
    resource disk "/opt/IBM/InformationServer/Server/Datasets" {pools ""}
    resource scratchdisk "/tmp" {pools ""}
  }
  node "Compute3"
  {
    fastname "taruspvt.itsosj.sanjose.ibm.com"
    pools ""
    resource disk "/opt/IBM/InformationServer/Server/Datasets" {pools ""}
    resource scratchdisk "/tmp" {pools ""}
  }
}
```

```
}  
}  
.....  
  Message:: main_program: APT configuration file:  
/home/dsadm/grid_jobdir/rowGenSeqFile_host2_g_26895.config.apt  
.....
```

Single job with a WebSphere MQ Connector stage

There are several Plug-ins (such as MQ) and stages (such as Distributed Transaction stage) in IBM Information Server that are server-based and therefore must be constrained to run on the conductor (front end) node because they cannot run on a compute node.

To demonstrate how this can be done, we use a DataStage job with a WebSphereMQConnectorPX stage that reads from a queue and writes its contents to a sequential file.

Note: In this case, the WebSphereMQConnectorPX could have executed on a compute node. Our objective here, however, is to show how to constrain its execution to the conductor node.

A DataStage job with a WebSphereMQConnectorPX stage that must be grid-enabled must be modified as follows:

- ▶ For single jobs and jobs including sequential file stages that must be grid-enabled, see Figure 2-32 on page 168 for information on adding grid environment parameters
- ▶ Constrain the Node pools and resource constraints to the conductor node in the WebSphereMQConnectorPX stage under the Advanced tab as shown in Figure 2-41 on page 176.

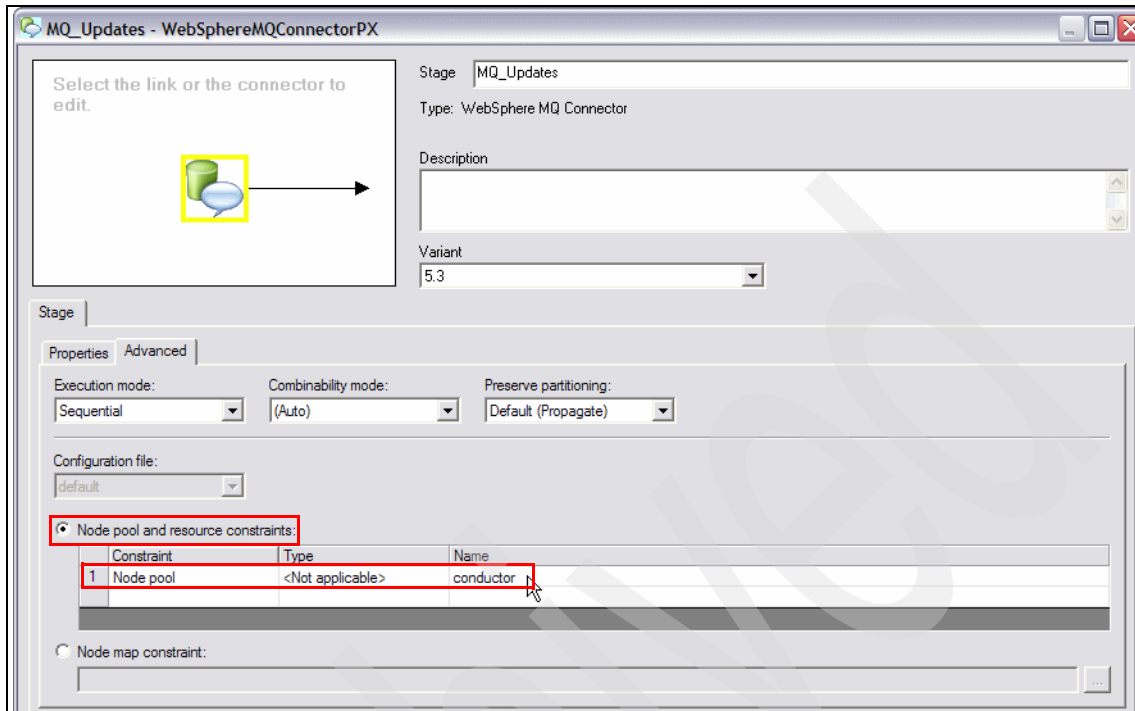


Figure 2-41 WebSphereMQConnectorPX stage conductor node pool constraint

Perform the following steps to grid-enable the MQ_g job that retrieves messages from a queue and writes them to a sequential file:

1. Modify the WebSphereMQConnectorPX stage to constrain the Node pools and resource constraints to the “conductor” node as shown in Figure 2-41. The default.appt file, shown in Example 2-109 on page 135, has to have a conductor node pool.
2. Modify the File attribute by prepending the file name with the string `#$APT_GRID_SEQFILE_HOST#` in the Target category under the Properties tab in the Input page of the Sequential File stage. See Figure 2-42 on page 178.

Figure 2-43 on page 179 shows the parameters defined for this MQ_g job. It shows an additional parameter `$APT_CONFIG_FILE` parameter with the prompt Configuration file with a default path name of `/opt/IBM/InformationServer/Server/Configurations/default.appt`. It is the non-grid parallel job configuration file that you can override in a non-grid environment.

`$APT_CONFIG_FILE` has no impact on a job that is grid-enabled.

3. Click the Run icon to run the job in Figure 2-44 on page 179 which brings up the Job Run Options window.
4. Modify the following parameter values in the Job Run Options window as shown in Figure 2-45 on page 180:
 - ENABLE - Use DYNAMIC_CONFIG_SCRIPT (NO/YES/OLD) value to YES
 - \$APT_GRID_SEQFILE_HOST with a prompt SEQFILE_HOST - First Host name identified by either Grid Engine or from IONODE names (job parameter returned) to blank
 - \$APT_GRID_SEQFILE_HOST2 with a prompt SEQFILE_HOST - Second Host name identified by either Grid Engine or from IONODE names (job parameter returned) to blank
 - COMPUTENODES - Number of Compute Nodes value to 2
 - PARTITIONS - Number of partitions per Node (compute) value to 2

Figure 2-46 on page 180 shows the successful execution of the job where 8 messages are retrieved and the same number are written to the sequential file.

The output of the DataStage and QualityStage Director log (Figure 2-47 on page 181) shows the following servers as the compute nodes:

- phoenixpvt.itsosj.sanjose.ibm.com
- taruspvt.itsosj.sanjose.ibm.com

The configuration file is MQ_g_19665.config.apl.

Note: Even though there is only one sequential file defined in this job and there are two variables (\$APT_GRID_SEQFILE_HOST and \$APT_GRID_SEQFILE_HOST2) defined in the job, the Director log shows the assignment of the first two nodes (machines) phoenixpvt.itsosj.sanjose.ibm.com and taruspvt.itsosj.sanjose.ibm.com to the \$APT_GRID_SEQFILE_HOST and \$APT_GRID_SEQFILE_HOST2 respectively.

The Event Detail log (Figure 2-48 on page 181) indicates the parameter values used in the execution of this job.

Figure 2-49 on page 182 through Figure 2-51 on page 184 show the Event Detail log for the MQ_Updates and Sequential_File_231 stages.

Figure 2-50 on page 183 and Figure 2-51 on page 184 show the nodepool (conductor) as constraining the MQ_Updates stage.

Figure 2-52 on page 184 shows the Event Detail log with the two compute nodes selected as being phoenixpvt.itsosj.sanjose.ibm.com and taruspvt.itsosj.sanjose.ibm.com.

Figure 2-53 on page 185 and Figure 2-54 on page 186 shows the SEQFILE Host(s) phoenixpvt.itsosj.sanjose.ibm.com and taruspvt.itsosj.sanjose.ibm.com and the contents of the generated MQ_g_19665.config.appt file which has four nodes in all: Two each corresponding to the phoenixpvt.itsosj.sanjose.ibm.com and taruspvt.itsosj.sanjose.ibm.com machines. The kazanpvt.itsosj.sanjose.ibm.com machine is the Conductor node.

The MQ_g job Director detail, time files, and **11monitor** command output is not shown here.

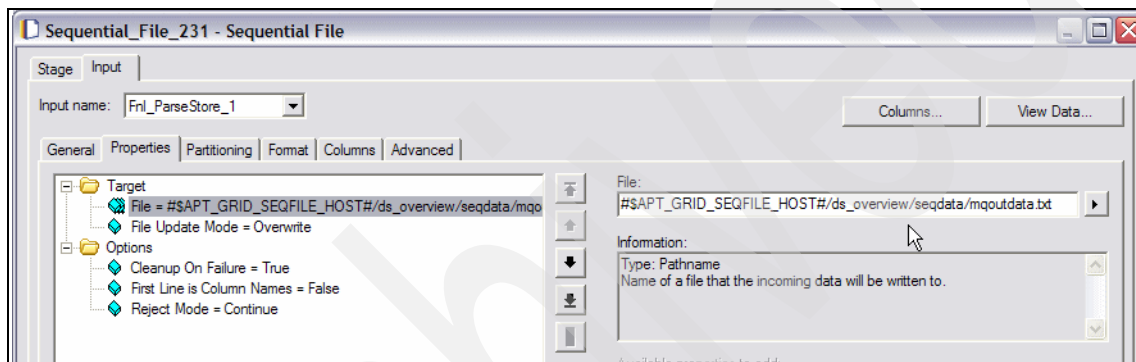


Figure 2-42 `$APT_GRID_SEQFILE_HOST` environment variable in Sequential file stage

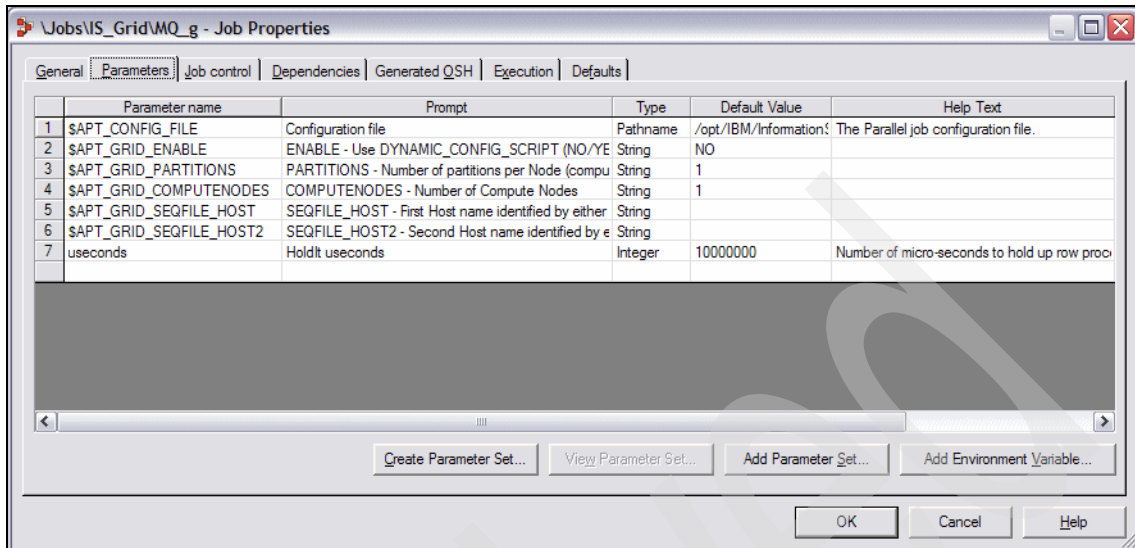


Figure 2-43 Parameters added to Job Properties for MQ_g

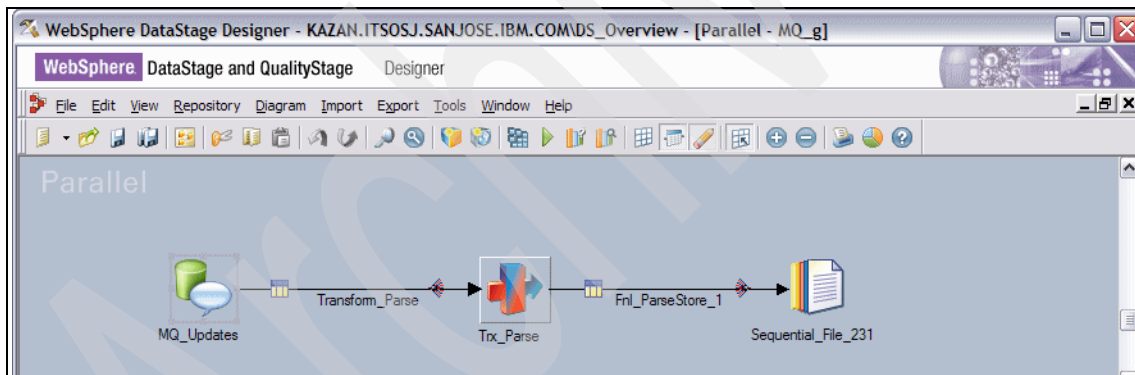


Figure 2-44 WebSphere MQ on “conductor” node, part 1 of 11

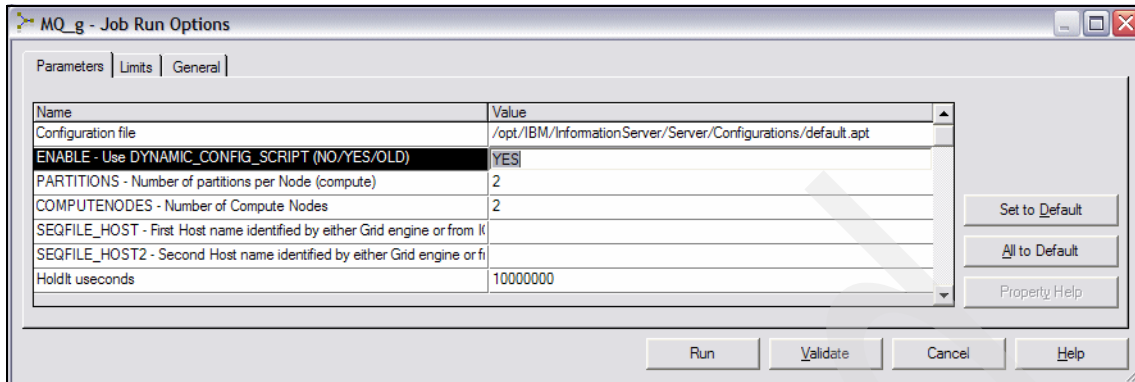


Figure 2-45 WebSphere MQ on “conductor” node, part 2 of 11

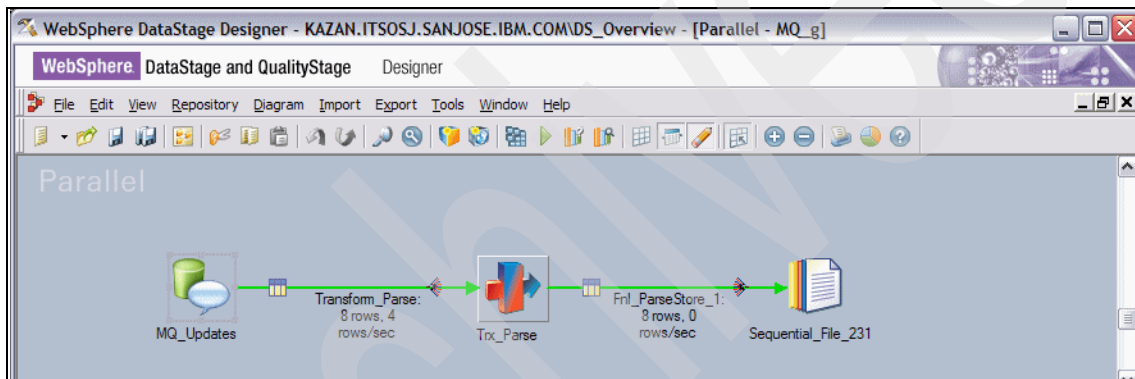


Figure 2-46 WebSphere MQ on “conductor” node, part 3 of 11

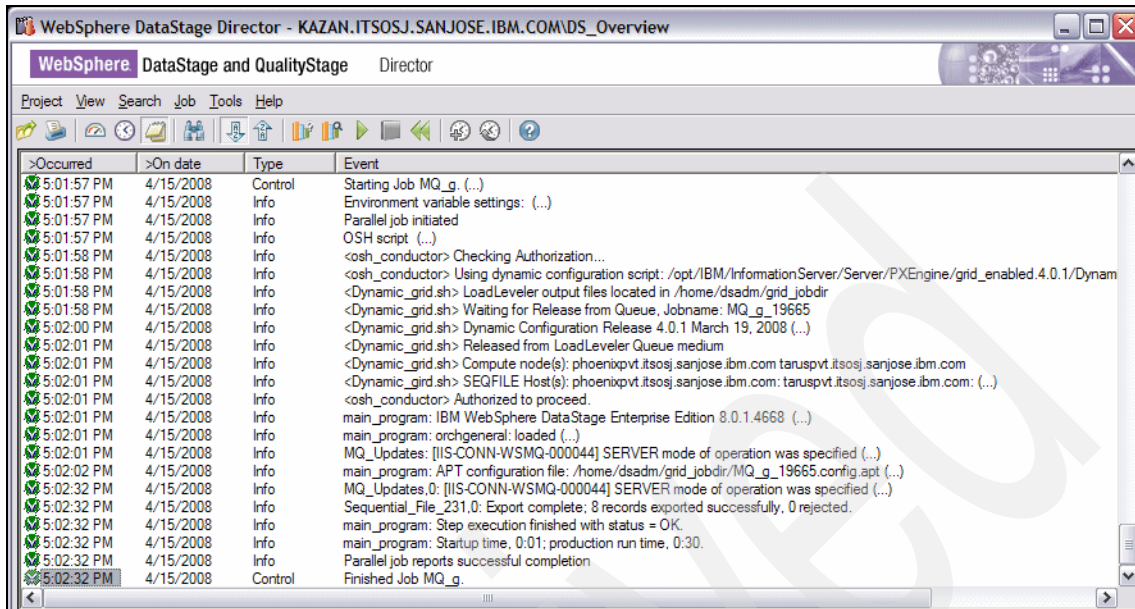


Figure 2-47 WebSphere MQ on “conductor” node, part 4 of 11

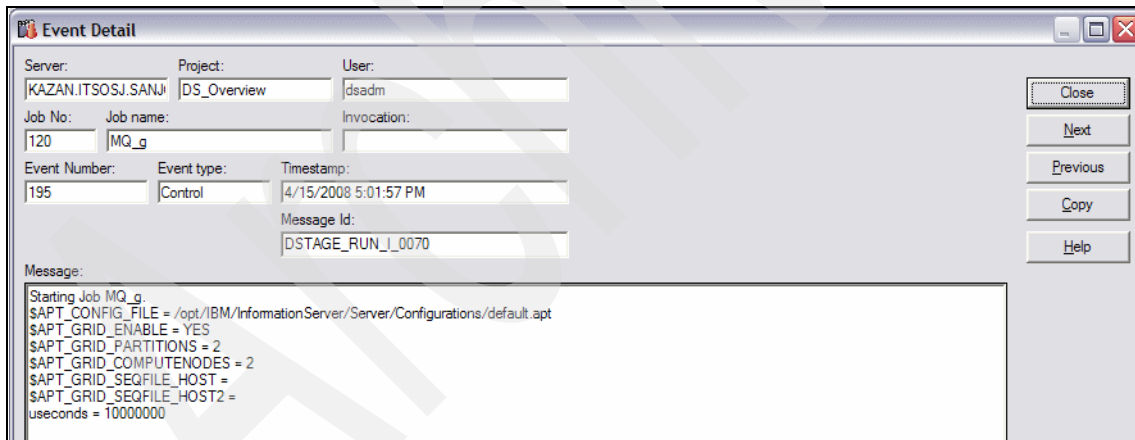


Figure 2-48 WebSphere MQ on “conductor” node, part 5 of 11

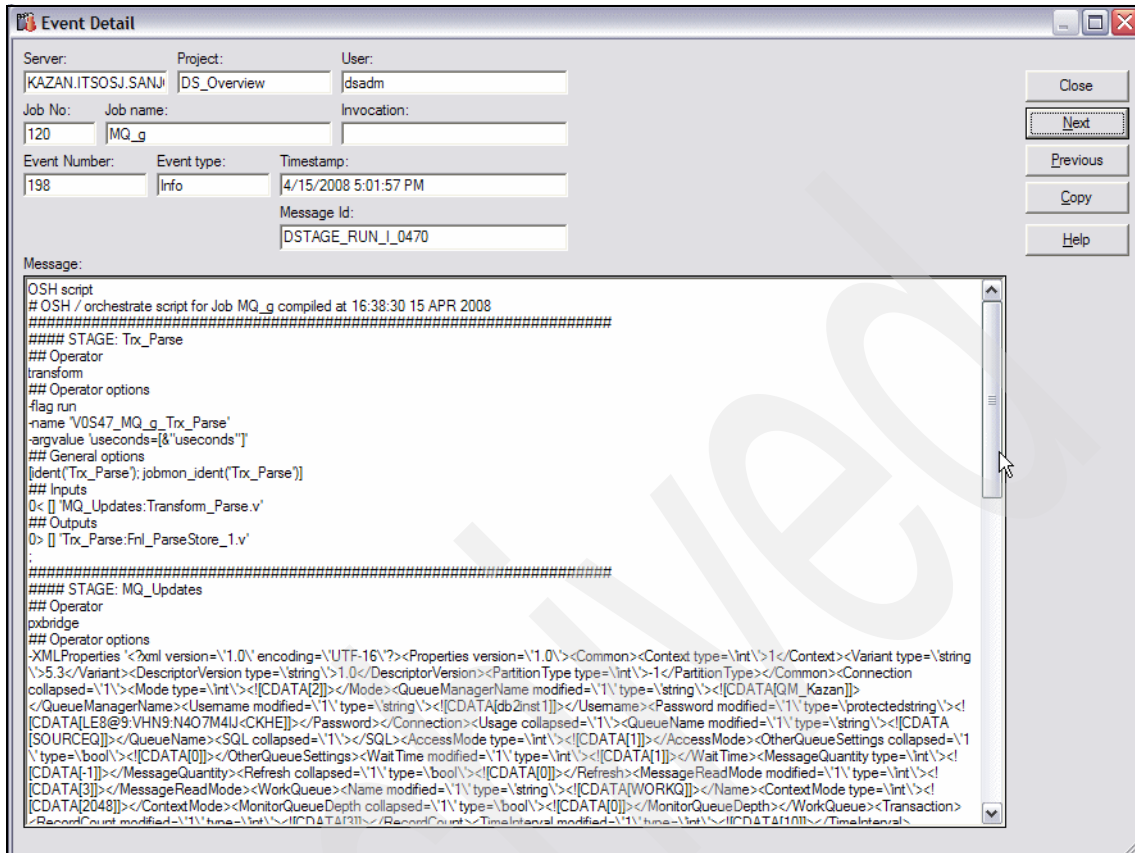


Figure 2-49 WebSphere MQ on “conductor” node, part 6 of 11

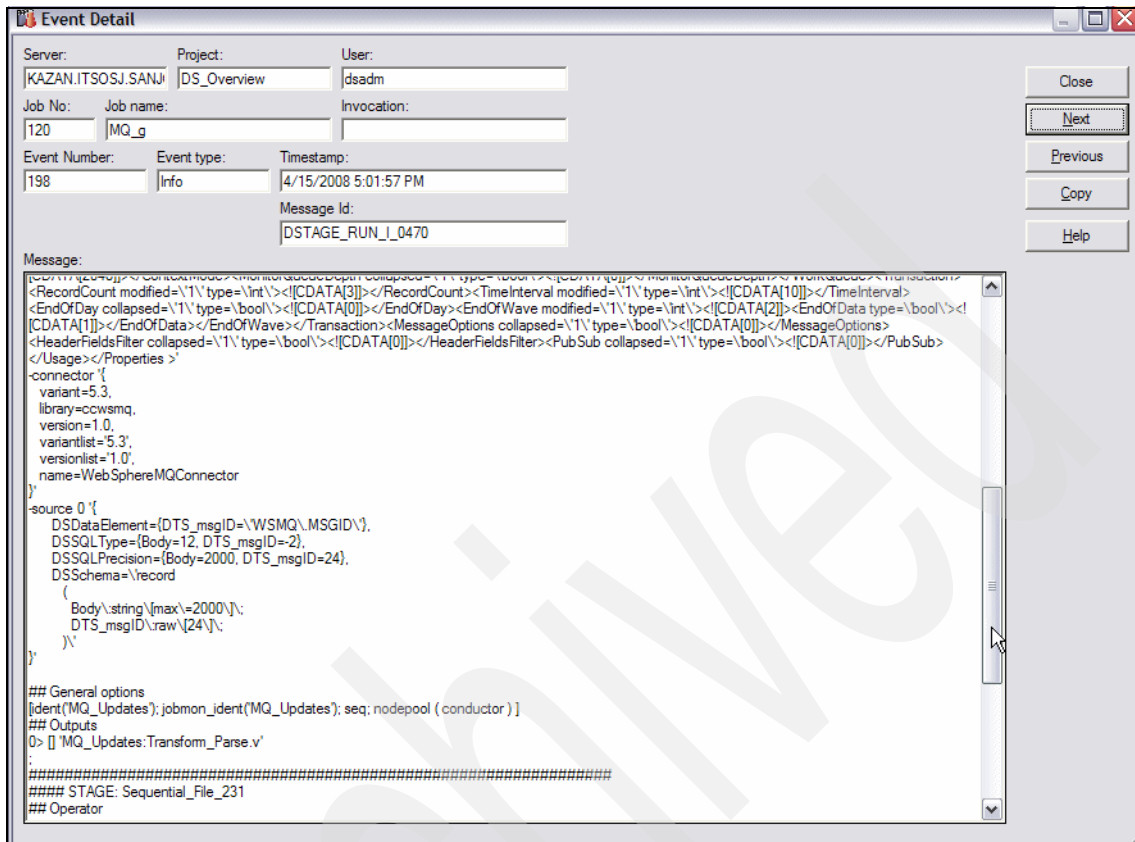


Figure 2-50 WebSphere MQ on “conductor” node, part 7 of 11

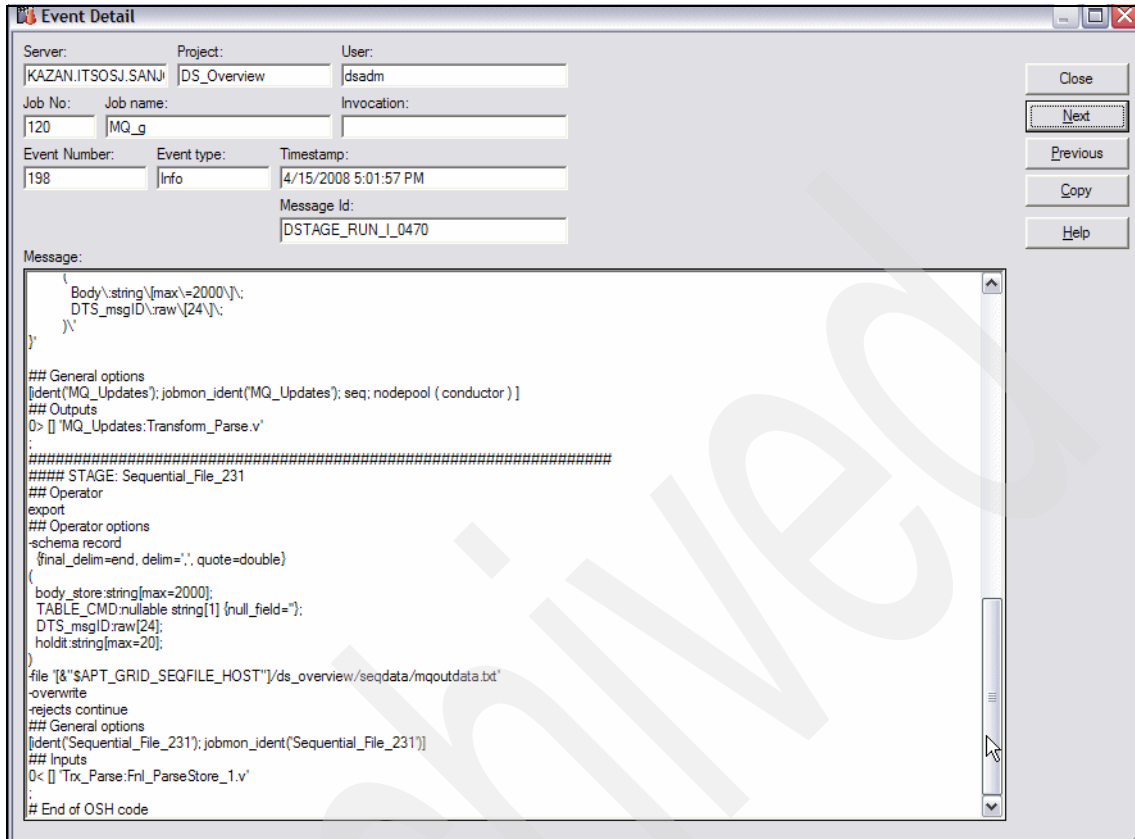


Figure 2-51 WebSphere MQ on “conductor” node, part 8 of 11

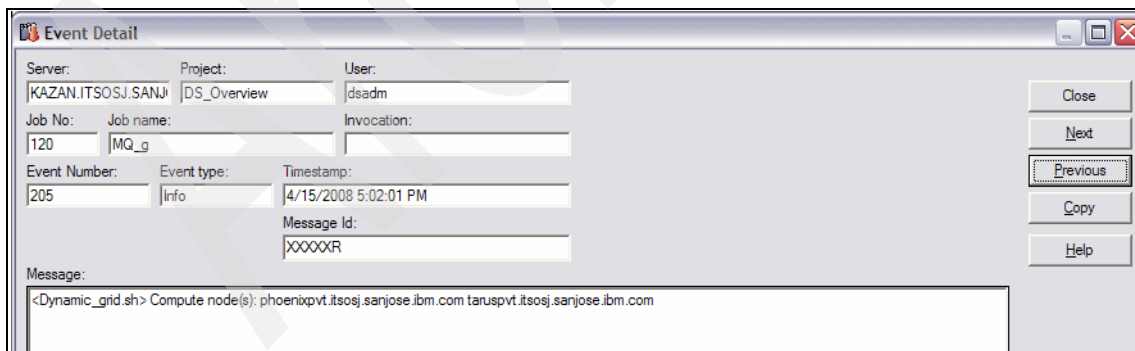


Figure 2-52 WebSphere MQ on “conductor” node, part 9 of 11

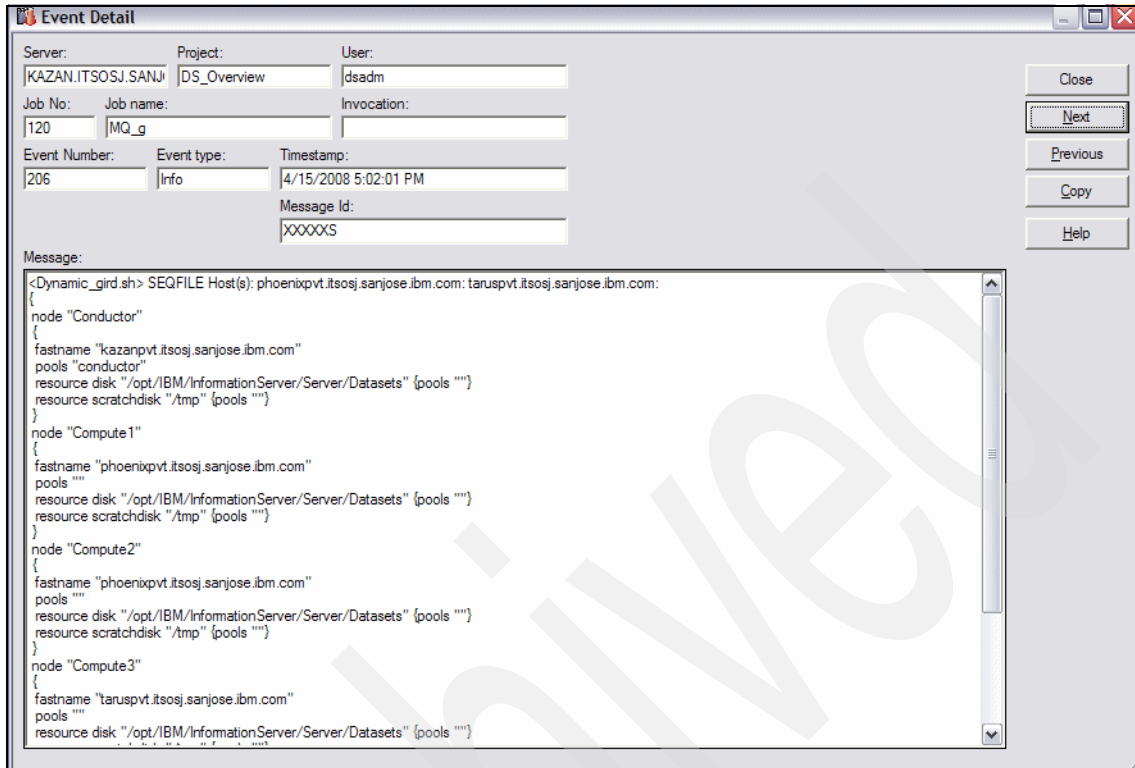


Figure 2-53 WebSphere MQ on "conductor" node, part 10 of 11

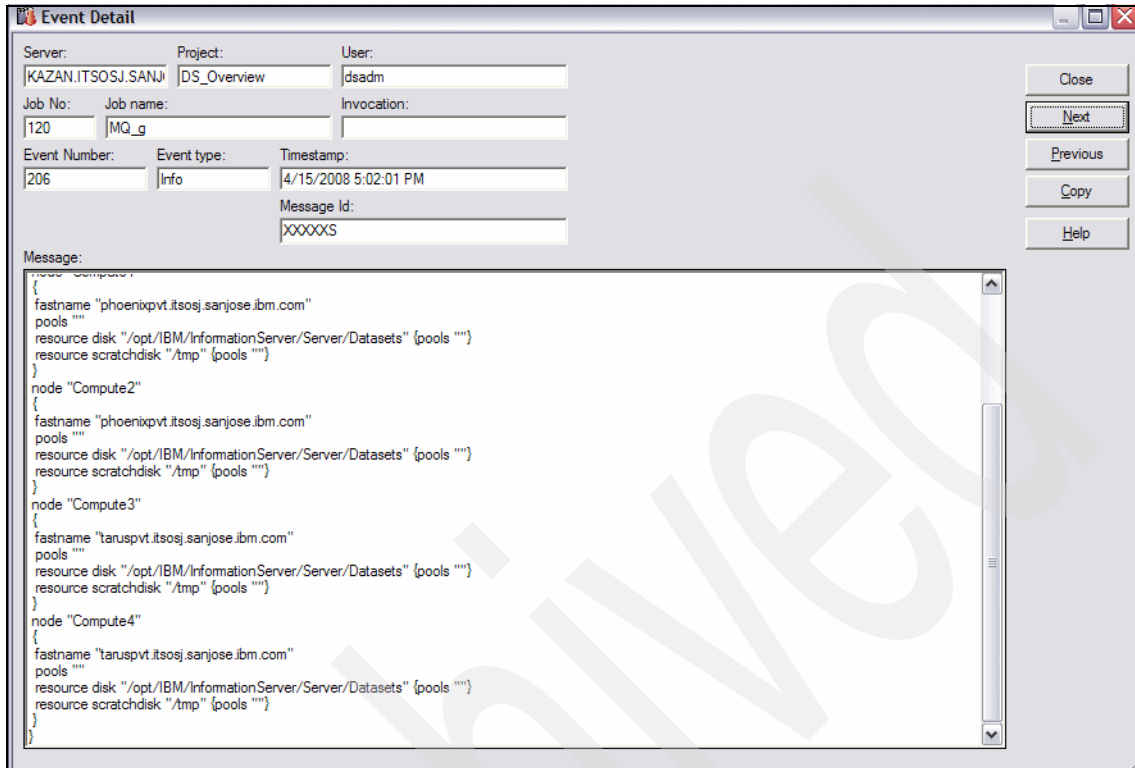


Figure 2-54 WebSphere MQ on “conductor” node, part 11 of 11

Job sequence

Grid-enabling a job sequence involves executing a `sequencer.sh` script provided by the toolkit that, by interacting with a resource manager, creates a dynamic configuration file based on the parameters supplied and the available resources. The dynamic configuration file is then passed to the individual jobs within the job sequence.

The key objective here is to identify the resources (in a dynamic configuration file) required for all the jobs in the job sequence only once, and then share this cost of creating the configuration file across all the jobs in the job sequence by reusing this dynamic configuration file within each job. Typically this makes sense when the individual jobs are short-running jobs as compared to the cost of creating a dynamic configuration file.

The sequencer.sh script accepts the following command line attributes:

```
-f <name of new apt_config_file>  
-c <number of compute nodes>  
-p <number of partitions per node>
```

The output of the sequencer.sh script is a single line that is space delimited with the following three values:

- ▶ \$APT_GRID_SEQFILE_HOST
- ▶ \$APT_GRID_SEQFILE_HOST2
- ▶ \$APT_CONFIG_FILE

Downstream job activity stages can then access these values using the expression: Field(<stagename>.\$CommandOutput,"",<fieldnum>)

A job sequence requires the following changes to exploit the grid environment:

1. Create an Execute Command Activity stage in the Designer client, and add parameters as shown in Figure 2-57 on page 191.
2. Specify sequencer.sh with its name in the Command field of the Execute Command Activity stage, and provide the parameters (-c #ComputeNodes# -p #Partitions#) in the Parameters field as shown in Figure 2-57 on page 191.
3. Add the parameters shown in Figure 2-41 on page 176 to each job in the job sequence, but set them as follows:
 - \$APT_GRID_ENABLE set to NO

Note: This must be set to NO because we do not want the individual jobs in the job sequence to repeat the process of reserving resources from the resource manager when the Execute Command Activity stage has already done so.

- \$APT_GRID_COMPUTENODES can be ignored
- \$APT_GRID_PARTITIONS can be ignored
- \$APT_GRID_SEQFILE_HOST set to Field(sequencer_sh.\$CommandOutput,"",1)

Note: This expression copies the first value (\$APT_GRID_SEQFILE_HOST) of the output from the sequencer.sh script output to the same variable of this job.

- \$APT_GRID_SEQFILE_HOST2 set to Field(sequencer_sh.\$CommandOutput," ,2)

Note: This expression copies the second value (\$APT_GRID_SEQFILE_HOST2) of the output from the sequencer.sh script output to the same variable of this job.

- \$APT_CONFIG_APT set to Field(sequencer_sh.\$CommandOutput," ,3)

Note: This expression copies the third value (\$APT_CONFIG_FILE) of the output from the sequencer.sh script output to the same variable of this job.

These settings ensure that the jobs in the sequence leverage the resources obtained by the sequencer.sh script and therefore the cost of allocating the resources is spread across all the jobs in the job sequence.

Figure 2-55 shows a job sequence rowGen_batch which calls the single jobs rowGenSleeper (this is similar to rowGenSeqFile except that the output is not written to a sequential file but to a Peek stage) and rowGenSeqFile (described in “Single job with a sequential file” on page 167).

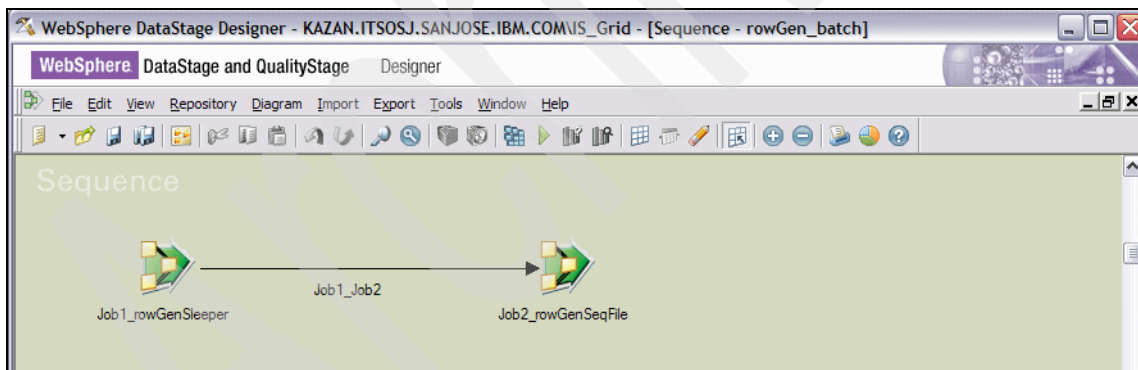


Figure 2-55 Multiple jobs rowGen_batch using job sequence

To migrate this rowGen_batch job sequence to a grid environment requires it to be modified as described earlier.

- Figure 2-56 on page 190, runMultipleJobs_host2_g, shows the modified job sequence to exploit the grid environment:

- The added Execute Command Activity stage named sequencer_sh in front of the rowGenSleeper_g and rowGenSeqFile_host2_g jobs.

The Command field of the Execute Command Activity stage is supplied the name of the sequencer.sh script, and the Parameters field is provided the parameters (-c #ComputeNodes# -p #Partitions#) as shown in Figure 2-57 on page 191.

- rowGenSleeper_g job is a grid-enabled version of “Single jobs” on page 148.

As required and shown in Figure 2-58 on page 191, parameters \$APT_GRID_ENABLE is set to NO and \$APT_CONFIG_FILE is set to the value expression Field(sequencer_sh.\$CommandOutput,””,3).

There are no \$APT_GRID_SEQFILE_HOST and \$APT_GRID_SEQFILE_HOST2 variable because this job does not have a sequential file.

- rowGenSeqFile_host2_g job is a grid-enabled version of “Single job with a sequential file” on page 167.

- Modify the File = attribute under the Property tab in the Sequential File stage tab to include ‘#\$APT_GRID_SEQFILE_HOST#’ preceding the filename as was shown in Figure 2-46 on page 180.

- As required and shown in Figure 2-59 on page 192, parameters \$APT_GRID_ENABLE is set to NO and \$APT_CONFIG_FILE is set to the value expression Field(sequencer_sh.\$CommandOutput,””,3). Because there is a sequential file in this job, the parameters \$APT_GRID_SEQFILE_HOST (value expression Field(sequencer_sh.\$CommandOutput,””,1)) and \$APT_GRID_SEQFILE_HOST2 (value expression Field(sequencer_sh.\$CommandOutput,””,2)) are set.

- Set the Job Properties of the rowGenSeqFile_host2_g as shown in Figure 2-60 on page 192. Set the ComputeNodes to 2 and Partitions to 3.
- Execute the runMultipleJobs_host2_g job sequence by clicking the run icon which brings up the Job Run Options window shown in Figure 2-61 on page 193. Let the values default as shown and click **Run**.

- Figure 2-62 on page 193 shows the Director log output of this job. The successful execution of the sequencer.sh script and the rowGenSleeper_g and rowGenSeqFile_host2_g jobs.

- ▶ Figure 2-63 on page 194 shows the event detail output with the parameters specified for this job sequence.
- ▶ Figure 2-64 on page 194 shows the event detail output with the successful execution of the sequencer.sh script and the three values in the output as follows:

luxorpvt.itsosj.sanjose.ibm.com: phoenix pvt.itsosj.sanjose.ibm.com:
/home/dsadm/grid_jobdir/runMultipleJobs_host2_g_27991.config.apt
- ▶ Figure 2-65 on page 195 shows the event detail output with the parameters used by the rowGenSleeper_g job after the value expression has been evaluated.
- ▶ Figure 2-66 on page 195 shows the event detail output with the parameters used by the rowGenSeqFile_host2_g job after the value expression has been evaluated.
- ▶ Figure 2-67 on page 196 and Figure 2-68 on page 197 show the Director output of the rowGenSleeper_g job.
- ▶ Figure 2-69 on page 198 and Figure 2-70 on page 199 show the dynamic configuration file used in the execution of this job. It shows a total of six compute nodes: Two (Compute1 and Compute2) on the luxorpvt.itsosj.sanjose.ibm.com machine, two (Compute3 and Compute4) on the phoenixpvt.itsosj.sanjose.ibm.com machine, and two (Compute5 and Compute6) on the taruspvt.itsosj.sanjose.ibm.com machine.
- ▶ Figure 2-71 on page 199 shows the Director output of the rowGenSeqFile_host2_g job.

The Director detail output for each of the jobs (runMultipleJobs_host2_g, rowGenSleeper_g, rowGenSeqFile_host2_g), time files, and **llmonitor** command output is not shown here.

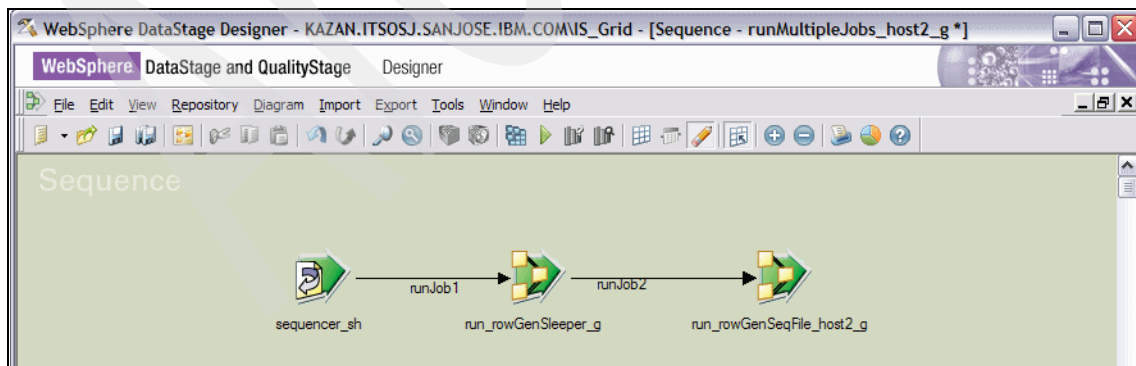


Figure 2-56 runMultipleJobs_host2_g job sequence

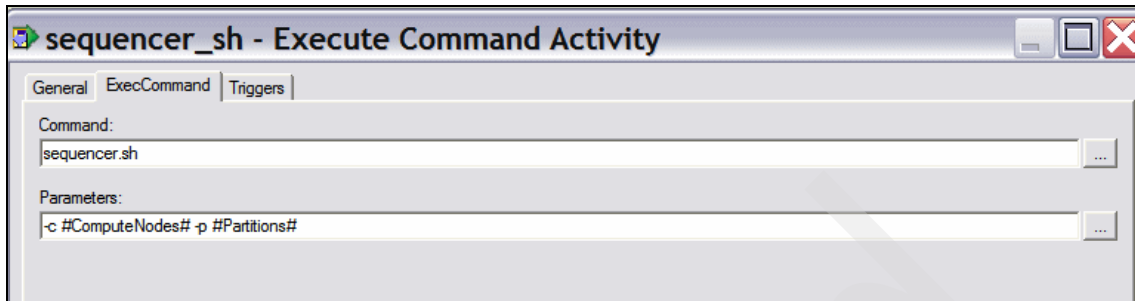


Figure 2-57 Execute Command Activity Exec Command configuration

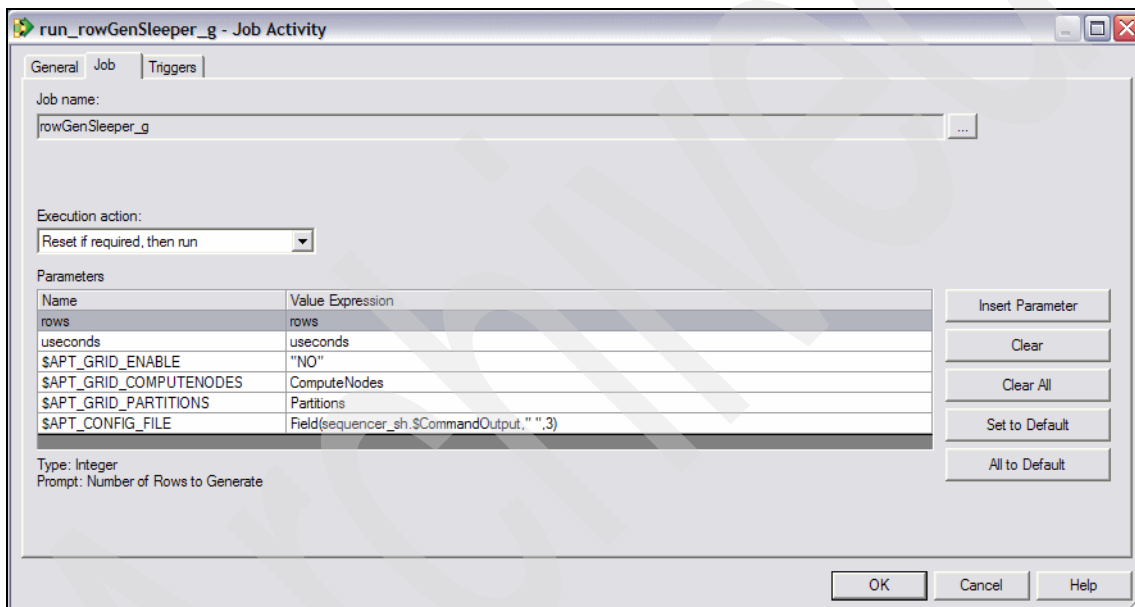


Figure 2-58 rowGenSleeper_g job Job Activity

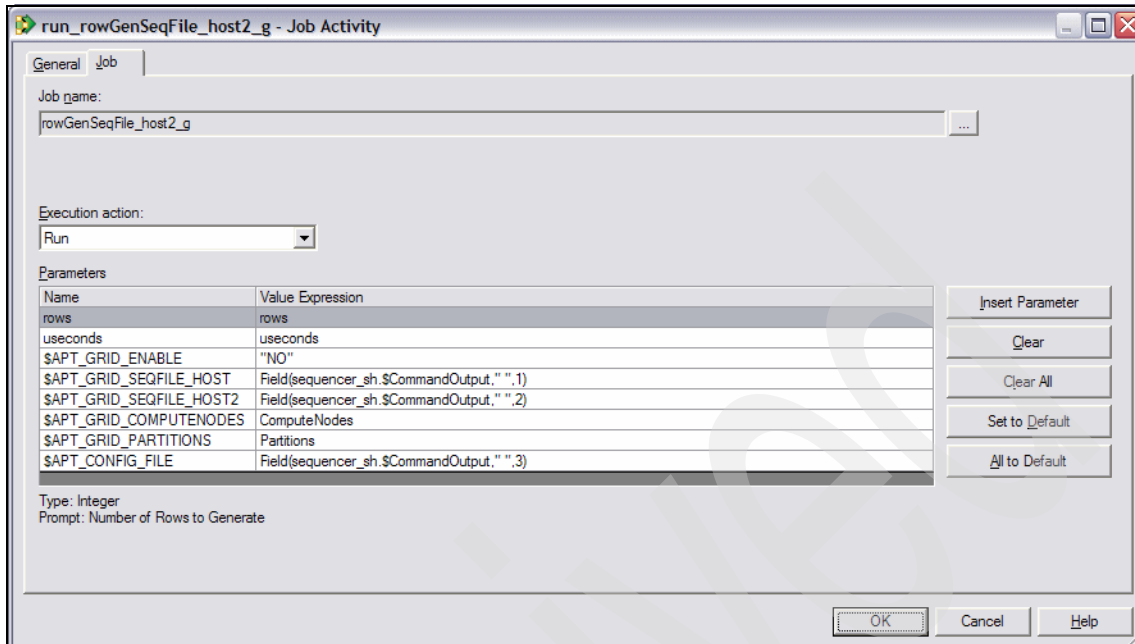


Figure 2-59 rowGenSeqFile_host2_g job Job Activity

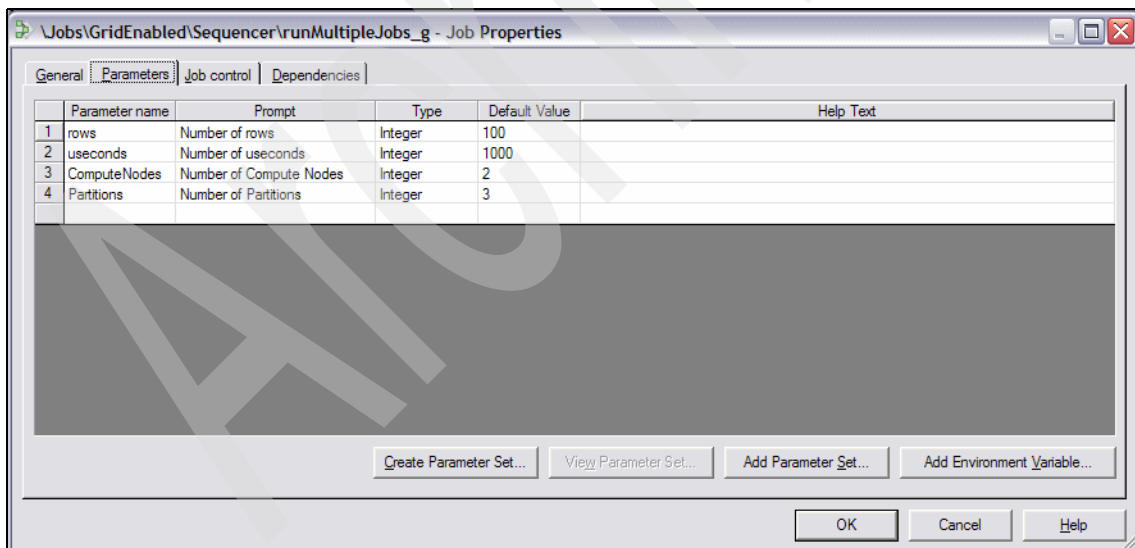


Figure 2-60 runMultipleJobs_g Job Properties

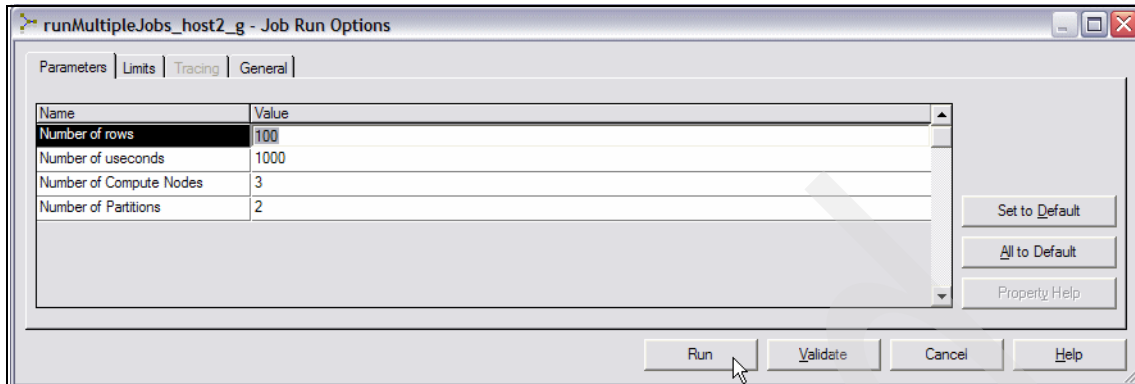


Figure 2-61 Multiple jobs (using Sequencer.sh), part 1 of 11

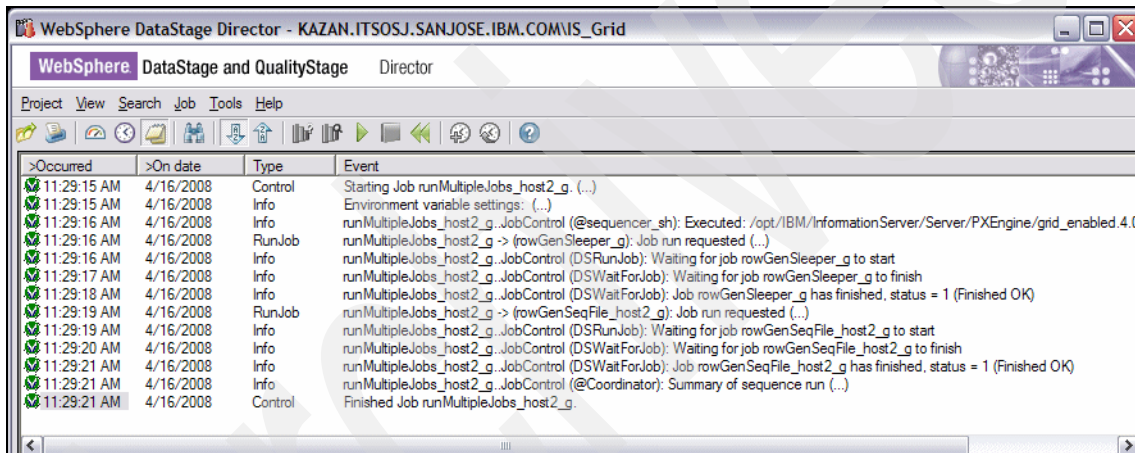


Figure 2-62 Multiple jobs (using Sequencer.sh), part 2 of 11

Event Detail			
Server:	Project:	User:	
KAZAN.ITSOSJ.SANJ	IS_Grid	dsadm	
Job No:	Job name:	Invocation:	
36	runMultipleJobs_host2_g		
Event Number:	Event type:	Timestamp:	
0	Control	4/16/2008 11:29:15 AM	
		Message Id:	
		DSTAGE_RUN_I_0070	
Message: Starting Job runMultipleJobs_host2_g. rows = 100 useconds = 1000 ComputeNodes = 3 Partitions = 2			

Close
 Next
 Previous
 Copy
 Help

Figure 2-63 Multiple jobs (using Sequencer.sh), part 3 of 11

Event Detail			
Server:	Project:	User:	
KAZAN.ITSOSJ.SANJ	IS_Grid	dsadm	
Job No:	Job name:	Invocation:	
36	runMultipleJobs_host2_g		
Event Number:	Event type:	Timestamp:	
2	Info	4/16/2008 11:29:16 AM	
		Message Id:	
		DSTAGE_RUN_I_0019	
Message: runMultipleJobs_host2_g..JobControl (@sequencer_sh): Executed: /opt/IBM/InformationServer/Server/PXEngine/grid_enabled.4.0.1/sequencer.sh -c 3 -p 2 Reply=0 Output from command ==>> luxorvrt.itsosj.sanjose.ibm.com: phoenixpvt.itsosj.sanjose.ibm.com: /home/dsadm/grid_jobdir/runMultipleJobs_host2_g_27991.config apt			

Close
 Next
 Previous
 Copy
 Help

Figure 2-64 Multiple jobs (using Sequencer.sh), part 4 of 11

Event Detail			
Server:	Project:	User:	
KAZAN.ITSOSJ.SANJi	IS_Grid	dsadm	
Job No:	Job name:	Invocation:	
36	runMultipleJobs_host2_g		
Event Number:	Event type:	Timestamp:	
3	RunJob	4/16/2008 11:29:16 AM	
		Message Id:	
		DSTAGE_RUN_I_0034	
<p>Message:</p> <pre>runMultipleJobs_host2_g -> (rowGenSleeper_g): Job run requested Mode (row/wam limits) = 0/0 Job Parameters --> rows=100 useconds=1000 \$APT_GRID_ENABLE=NO \$APT_GRID_COMPUTENODES=3 \$APT_GRID_PARTITIONS=2 \$APT_CONFIG_FILE=/home/dsadm/grid_jobdir/runMultipleJobs_host2_g_27991.config.apt DSJobController=runMultipleJobs_host2_g</pre>			

Figure 2-65 Multiple jobs (using Sequencer.sh), part 5 of 11

Event Detail			
Server:	Project:	User:	
KAZAN.ITSOSJ.SANJi	IS_Grid	dsadm	
Job No:	Job name:	Invocation:	
36	runMultipleJobs_host2_g		
Event Number:	Event type:	Timestamp:	
7	RunJob	4/16/2008 11:29:19 AM	
		Message Id:	
		DSTAGE_RUN_I_0034	
<p>Message:</p> <pre>runMultipleJobs_host2_g -> (rowGenSeqFile_host2_g): Job run requested Mode (row/wam limits) = 0/0 Job Parameters --> rows=100 useconds=1000 \$APT_GRID_ENABLE=NO \$APT_GRID_SEQFILE_HOST=luxorpxvt.itsosj.sanjose.ibm.com: \$APT_GRID_SEQFILE_HOST2=phoenixpxvt.itsosj.sanjose.ibm.com: \$APT_GRID_COMPUTENODES=3 \$APT_GRID_PARTITIONS=2 \$APT_CONFIG_FILE=/home/dsadm/grid_jobdir/runMultipleJobs_host2_g_27991.config.apt DSJobController=runMultipleJobs_host2_g</pre>			

Figure 2-66 Multiple jobs (using Sequencer.sh), part 6 of 11

WebSphere DataStage Director - KAZAN.ITSOSJ.SANJOSE.IBM.COM\NIS_Grid			
WebSphere DataStage and QualityStage Director			
Project View Search Job Tools Help			
>Occurred	>On date	Type	Event
✓ 11:29:18 AM	4/16/2008	Info	PeekRows,5: c2:1029 c1:3333333333 c3:333333333
✓ 11:29:18 AM	4/16/2008	Info	PeekRows,1: c2:1031 c1:5555555555 c3:555555555555
✓ 11:29:18 AM	4/16/2008	Info	PeekRows,4: c2:1034 c1:8888888888 c3:888888
✓ 11:29:18 AM	4/16/2008	Info	PeekRows,5: c2:1035 c1:9999999999 c3:9
✓ 11:29:18 AM	4/16/2008	Info	PeekRows,2: c2:1032 c1:6666666666 c3:666666666666
✓ 11:29:18 AM	4/16/2008	Info	PeekRows,0: c2:1030 c1:4444444444 c3:444444444444
✓ 11:29:18 AM	4/16/2008	Info	PeekRows,3: c2:1027 c1:1111111111 c3:
✓ 11:29:18 AM	4/16/2008	Info	PeekRows,1: c2:1037 c1:BBBBBBBBBB c3:BBBBBBBBBBBBBB
✓ 11:29:18 AM	4/16/2008	Info	PeekRows,4: c2:1040 c1:EEEEEEEEEE c3:EEEEEEEEEEEEEEEE
✓ 11:29:18 AM	4/16/2008	Info	PeekRows,5: c2:1041 c1:FFFFFFFF c3:FFFFF
✓ 11:29:18 AM	4/16/2008	Info	PeekRows,2: c2:1038 c1:CCCCCCCC c3:CCCCCCC
✓ 11:29:18 AM	4/16/2008	Info	PeekRows,0: c2:1036 c1:AAAAAAAA c3:AAAA
✓ 11:29:18 AM	4/16/2008	Info	PeekRows,3: c2:1033 c1:7777777777 c3:7777777777
✓ 11:29:18 AM	4/16/2008	Info	PeekRows,1: c2:1043 c1:HHHHHHHH c3:HHHHHHHHHHHH
✓ 11:29:18 AM	4/16/2008	Info	PeekRows,4: c2:1046 c1:KKKKKKKK c3:KKKKKKKKKKKK
✓ 11:29:18 AM	4/16/2008	Info	PeekRows,5: c2:1047 c1:LLLLLLLL c3:LLLLLLLLLLLLLLLL
✓ 11:29:18 AM	4/16/2008	Info	PeekRows,2: c2:1044 c1:IIIIII c3:IIII
✓ 11:29:18 AM	4/16/2008	Info	PeekRows,0: c2:1042 c1:GGGGGGGG c3:GGGGGGGGGGGG
✓ 11:29:18 AM	4/16/2008	Info	PeekRows,3: c2:1039 c1:DDDDDDDD c3:DDDDDDDDDDDD
✓ 11:29:18 AM	4/16/2008	Info	PeekRows,1: c2:1049 c1:NNNNNNNN c3:NNNNNNNNNNNN
✓ 11:29:18 AM	4/16/2008	Info	PeekRows,4: c2:1052 c1:QQQQQQQQ c3:QQQQQQQQQQQQ
✓ 11:29:18 AM	4/16/2008	Info	PeekRows,5: c2:1053 c1:RRRRRRRR c3:RRR
✓ 11:29:18 AM	4/16/2008	Info	PeekRows,2: c2:1050 c1:OOOOOOOO c3:OOOOOOOO
✓ 11:29:18 AM	4/16/2008	Info	PeekRows,0: c2:1048 c1:MMMMMMMM c3:MMMMMMMMMMMM
✓ 11:29:18 AM	4/16/2008	Info	PeekRows,3: c2:1045 c1:JJJJJJJJ c3:
✓ 11:29:18 AM	4/16/2008	Info	PeekRows,1: c2:1055 c1:TTTTTTTT c3:TTTTTTTTTTTTTT
✓ 11:29:18 AM	4/16/2008	Info	PeekRows,4: c2:1058 c1:WWWWWWWW c3:WWWWWWWWWWWW
✓ 11:29:18 AM	4/16/2008	Info	PeekRows,5: c2:1059 c1:XXXXXXXX c3:XXXXXXXXXXXXXXXX
✓ 11:29:18 AM	4/16/2008	Info	PeekRows,2: c2:1056 c1:UUUUUUUU c3:UU
✓ 11:29:18 AM	4/16/2008	Info	PeekRows,0: c2:1054 c1:SSSSSSSS c3:SSSSSSSSSSSS
✓ 11:29:18 AM	4/16/2008	Info	PeekRows,3: c2:1051 c1:PPPPPPPP c3:PPPPPPPPPPPP (...)
✓ 11:29:18 AM	4/16/2008	Info	main_program: Step execution finished with status = OK.
✓ 11:29:18 AM	4/16/2008	Info	main_program: Startup time, 0:00; production run time, 0:00.
✓ 11:29:18 AM	4/16/2008	Info	Parallel job reports successful completion
✓ 11:29:18 AM	4/16/2008	Control	Finished Job rowGenSleeper_g.
✓ 11:29:18 AM	4/16/2008	RunJob	(runMultipleJobs_host2_g) <- rowGenSleeper_g: Job under control finished.
✓ 11:31:35 AM	4/16/2008	Reset	Log prior to last run cleared by user

Figure 2-68 Multiple jobs (using Sequencer.sh), part 8 of 11

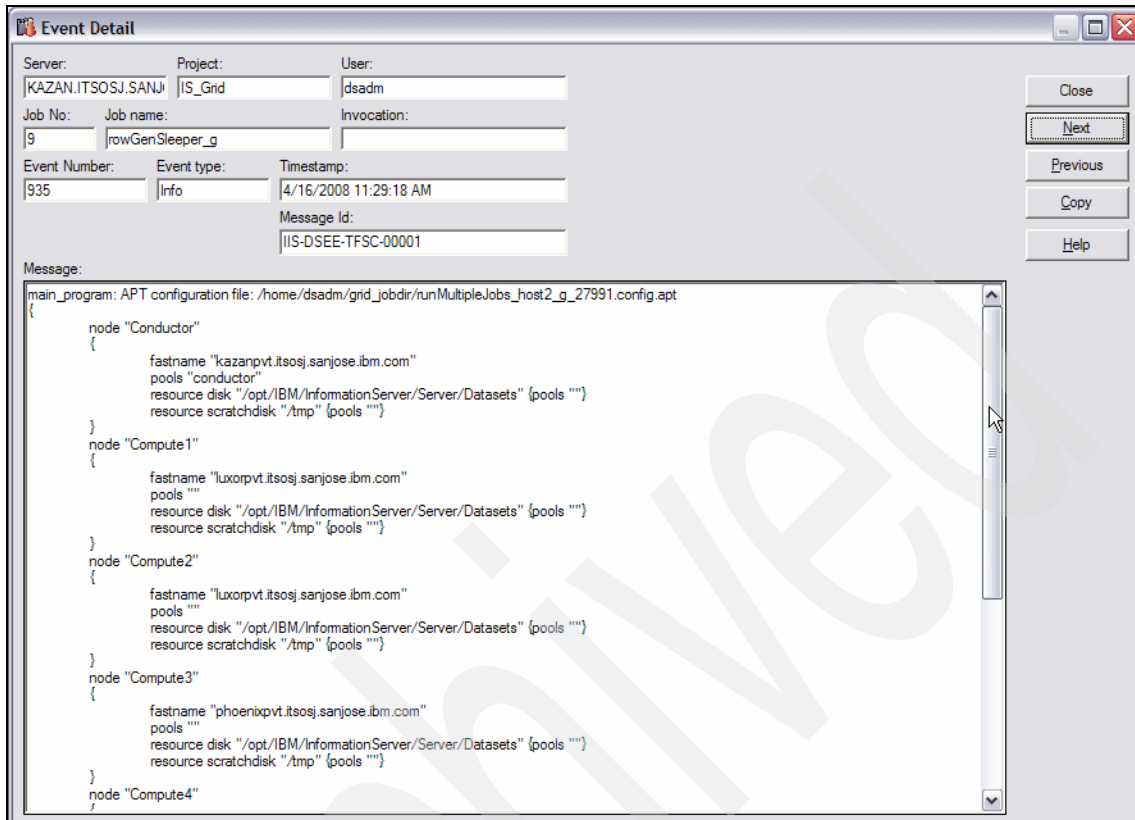


Figure 2-69 Multiple jobs (using Sequencer.sh), part 9 of 11

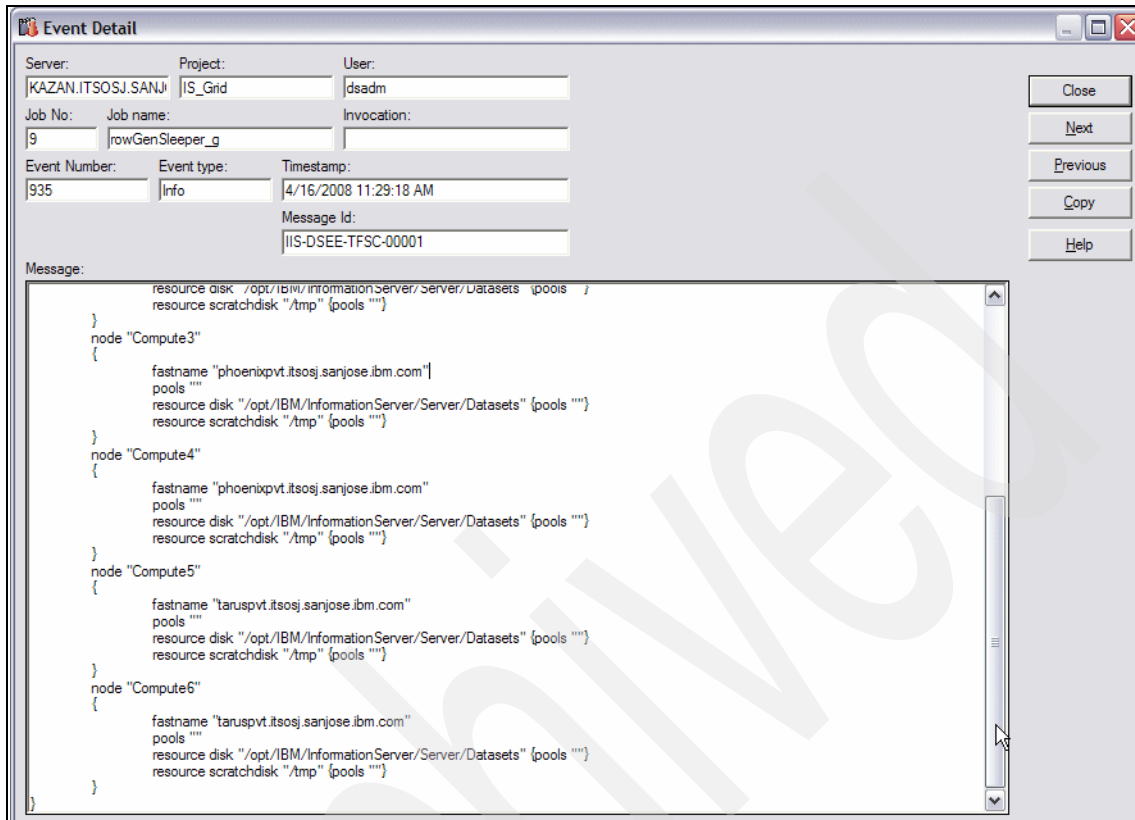


Figure 2-70 Multiple jobs (using Sequencer.sh), part 10 of 11

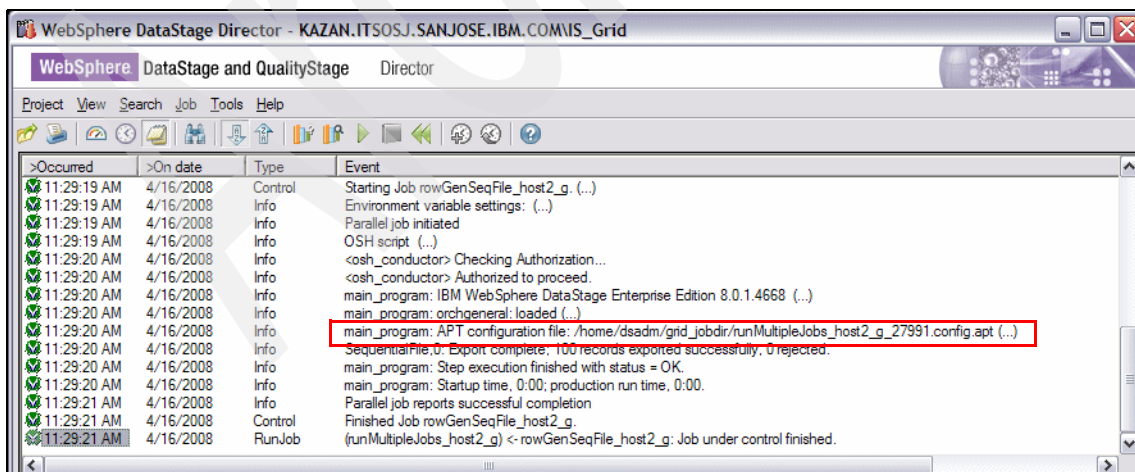


Figure 2-71 Multiple jobs (using Sequencer.sh), part 11 of 11

2.7 Step 3: Managing the grid environment

This section describes the procedures for performing the following routine tasks:

- ▶ Jobs rejected during submission
- ▶ Jobs in the Workload Scheduler LoadLeveler queue
- ▶ Why a job has still not been released from the queue
- ▶ Nodes' status including use of the individual compute nodes
- ▶ Cancel a waiting job from the queue
- ▶ Add/remove nodes

Performance management tasks are beyond the scope of this Redbooks publication.

2.7.1 Jobs rejected during submission

When you submit a job and its gets rejected, the message returned to the submitter depends upon the error such as “A valid queue must be defined. Valid queues are ...”, or “Value of \$APT_GRID_COMPUTENODES is greater than MaxNodesPerJob (N)”. To determine the cause of the rejection, you can view Event Detail in the Director log. We simulated three scenarios as follows:

- ▶ Submitted a job with an invalid message queue “No_Class1”.

This is shown in Figure 2-72 and Figure 2-73 on page 201.

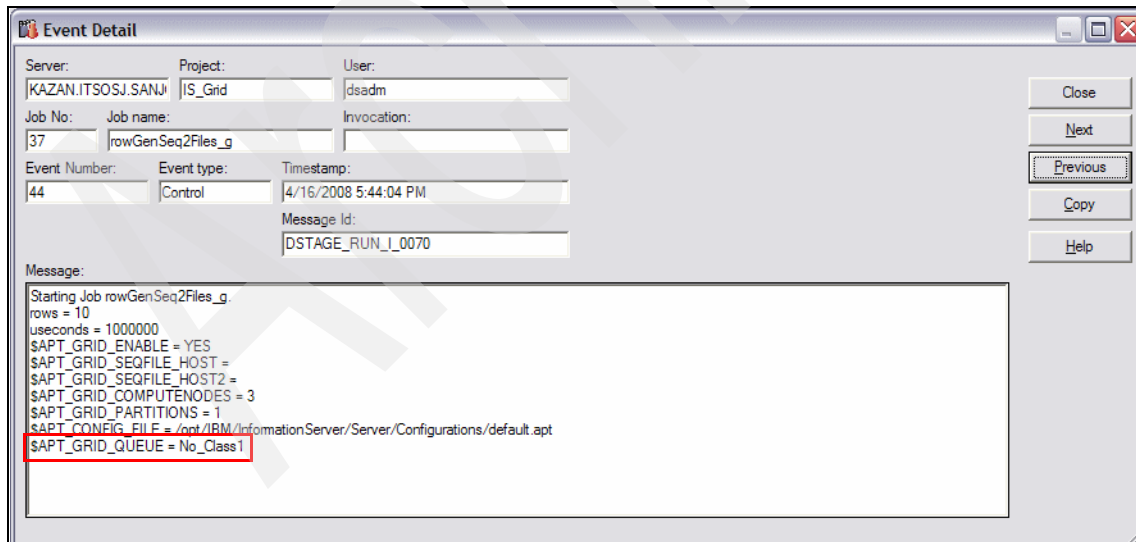


Figure 2-72 Jobs rejected during submission: Invalid message queue, part 1 of 2

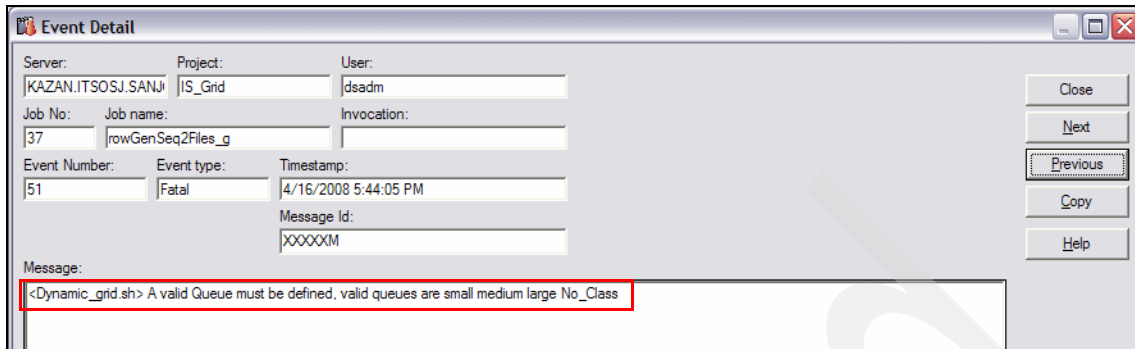


Figure 2-73 Jobs rejected during submission: Invalid message queue, part 2 of 2

- Submitted a job with the wrong case (all lower-case) “no_class” instead of the correct mixed case “No_Class.” Case is relevant.

This is shown in Figure 2-74 and Figure 2-75 on page 202.

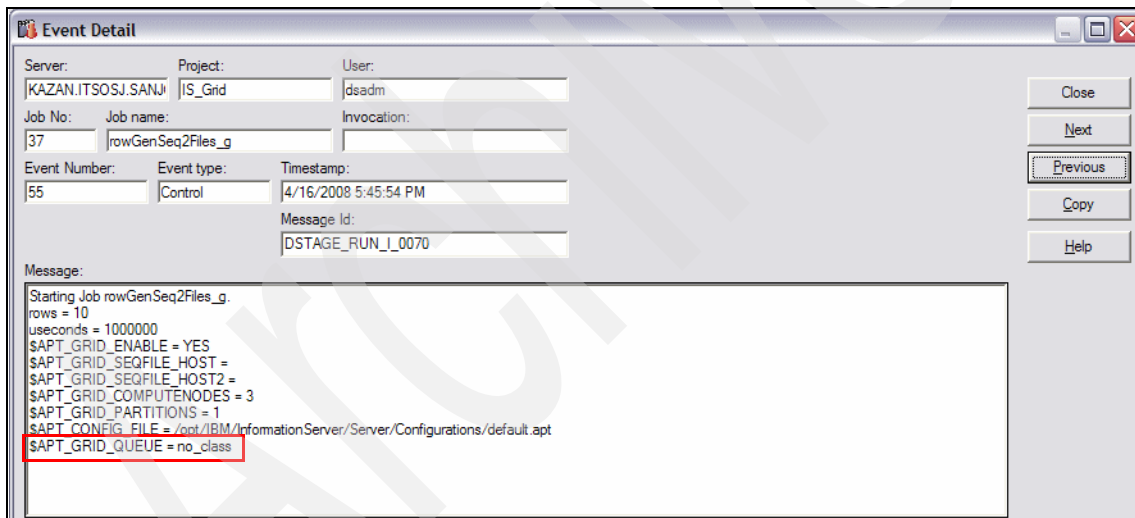


Figure 2-74 Jobs rejected during submission: Invalid message queue (wrong case), part 1 of 2

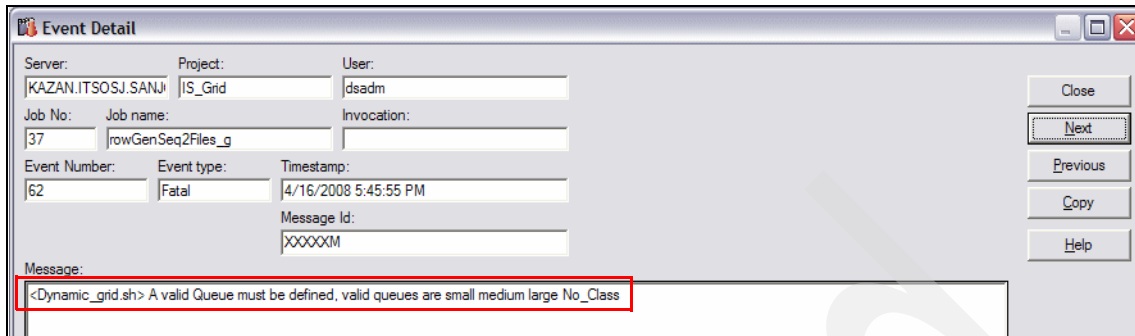


Figure 2-75 Jobs rejected during submission: Invalid message queue (wrong case), part 2 of 2

- ▶ Submitted a job with the number of compute nodes (four) that exceeded the maximum permitted number (three).

This is shown in Figure 2-76 and Figure 2-77 on page 203.

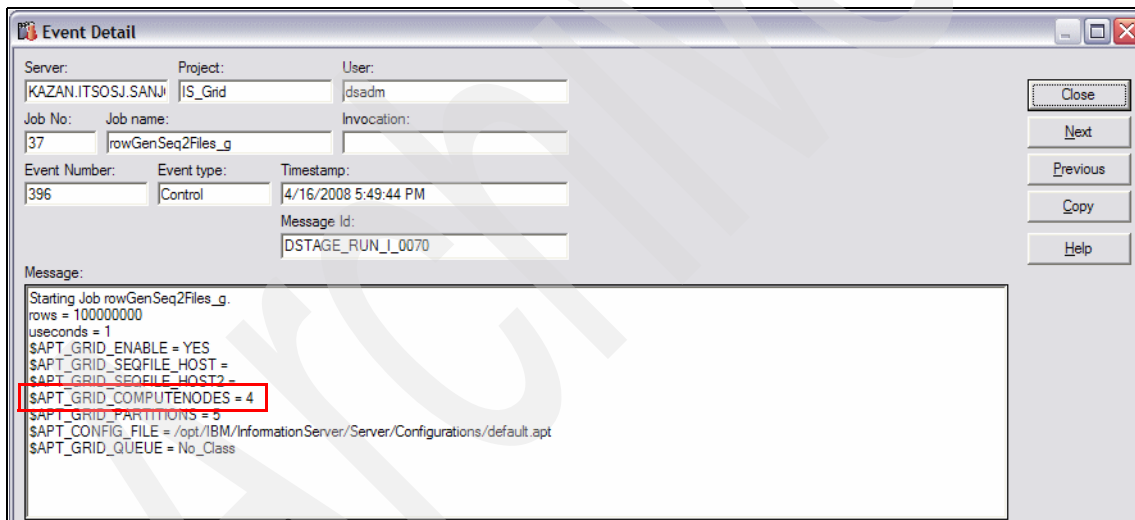


Figure 2-76 Jobs rejected during submission: Exceeds maximum nodes threshold, part 1 of 2

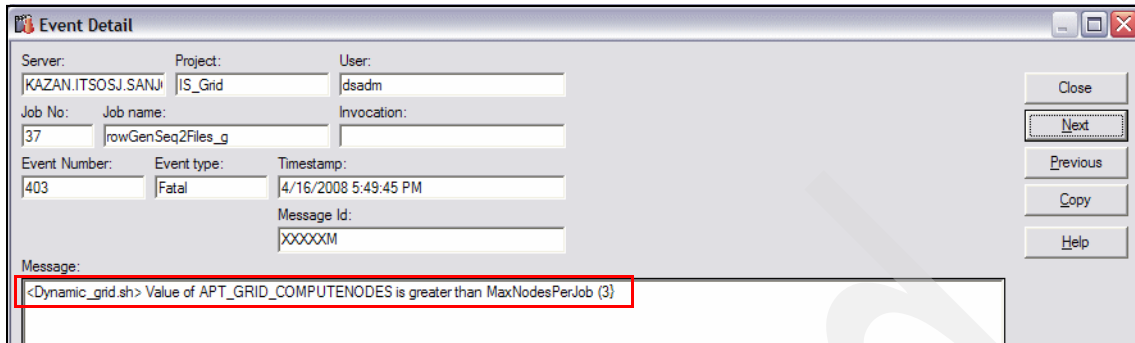


Figure 2-77 Jobs rejected during submission: Exceeds maximum nodes threshold, part 2 of 2

2.7.2 Jobs in the Workload Scheduler LoadLeveler queue

Grid-enabled jobs are submitted into the resource manager (Workload Scheduler LoadLeveler) queue and released for execution when the resources required for the job becomes available.

- The **llq** command of LoadLeveler can be used to obtain information about jobs in the LoadLeveler queues. Example 2-128 shows two jobs orion.84.0 and orion.85.0, both in the No_Class queue: One waiting and one running on tarus.

Example 2-128 llq command (Workload Scheduler LoadLeveler)

```
[dsadm@orion ~]$ llq
```

Id	Owner	Submitted	ST	PRI	Class
Running On					

orion.84.0	dsadm	4/16 17:51 R	50		No_Class
tarus					
orion.85.0	dsadm	4/16 17:55 I	50		No_Class

2 job step(s) in queue, 1 waiting, 0 pending, 1 running, 0 held, 0 preempted

- The **listq** command comes with the grid toolkit (and is resource manager independent) and provides more or less the same information as shown in Example 2-129.

Example 2-129 listq command (Grid toolkit resource manager command)

```
[dsadm@orion ~]$ listq
Step Id          Job Name          ST PRI Class
Running On
-----
-----
orion.85.0        rowGenSeqFile_28910 I  50  No_Class
orion.84.0        rowGenSeq2Files_g_27 R  50  No_Class
tarus

2 job step(s) in queue, 1 waiting, 0 pending, 1 running, 0 held, 0
preempted
```

2.7.3 Why a job has still not been released from the queue

When a submitted job does not get released from the queue, you can determine the cause by first determining the Step ID of the waiting job and then issuing the **listq <Step Id>** as follows:

- Example 2-130 lists all the jobs in the LoadLeveler queue and their status using the **listq** command. The Step ID of the waiting job is orion.85.0

Example 2-130 List contents of the Workload Scheduler LoadLeveler queue using the listq command

```
[dsadm@orion ~]$ listq
Step Id          Job Name          ST PRI Class
Running On
-----
-----
orion.85.0        rowGenSeqFile_28910 I  50  No_Class
orion.84.0        rowGenSeq2Files_g_27 R  50  No_Class
tarus

2 job step(s) in queue, 1 waiting, 0 pending, 1 running, 0 held, 0
preempted
```

- Example 2-131 shows the output of the **listq orion.85.0** command.
The reason for the job not being released is that there were no initiators for the No_Class on the nodes as the highlighted text shows.

Example 2-131 Getting more details on waiting jobs (output of listq orion.85.0)

```
[dsadm@orion ~]$ listq orion.85.0
.....
===== Job Step orion.itsosj.sanjose.ibm.com.85.0
=====
          Job Step Id: orion.itsosj.sanjose.ibm.com.85.0
          Job Name: rowGenSeqFile_28910
          Step Name: 0
    Structure Version: 10
          Owner: dsadm
        Queue Date: Wed 16 Apr 2008 05:55:50 PM PDT
          Status: Idle
.....
-----
Node
----

Name      :
Requirements : (Arch == "i386") && (OpSys == "Linux2")
Preferences :
Node minimum : 1
Node maximum : 1
Node actual  : 0
Allocated Hosts :

Master Task
-----

Executable : /home/dsadm/grid_jobdir/rowGenSeqFile_28910.sh
Exec Args  :
Num Task Inst: 0
Task Instance:

Task
----

Num Task Inst: 0
Task Instance:
```

```
===== EVALUATIONS FOR JOB STEP  
orion.itsosj.sanjose.ibm.com.85.0 =====
```

```
The class of this job step is "No_Class".  
Total number of available initiators of this class on all machines  
in the cluster: 0  
Minimum number of initiators of this class required by job step: 1  
The number of available initiators of this class is not sufficient  
for this job step.  
Not enough resources to start now:  
This step is top-dog.  
Considered at: Wed Apr 16 18:08:13 2008  
Will start by: Wed Apr 16 18:21:11 2008
```

2.7.4 Node status including use of the individual compute nodes

The status and use of the individual nodes in the grid environment may be viewed using a monitoring tool such as ganglia.

Figure 2-78 on page 207 and Figure 2-79 on page 208 provide a high level overview of the status of all the nodes in the grid environment.

- ▶ Figure 2-78 on page 207 shows the following information:
 - Three up hosts and one down host
 - Total of 30 CPUs in the grid environment
- ▶ Figure 2-79 on page 208 shows the specific nodes that are down (luxor.itsosj.sanjose.ibm.com) and up (phoenix.itsosj.sanjose.ibm.com, orion.itsosj.sanjose.ibm.com, and tarus.itsosj.sanjose.ibm.com)

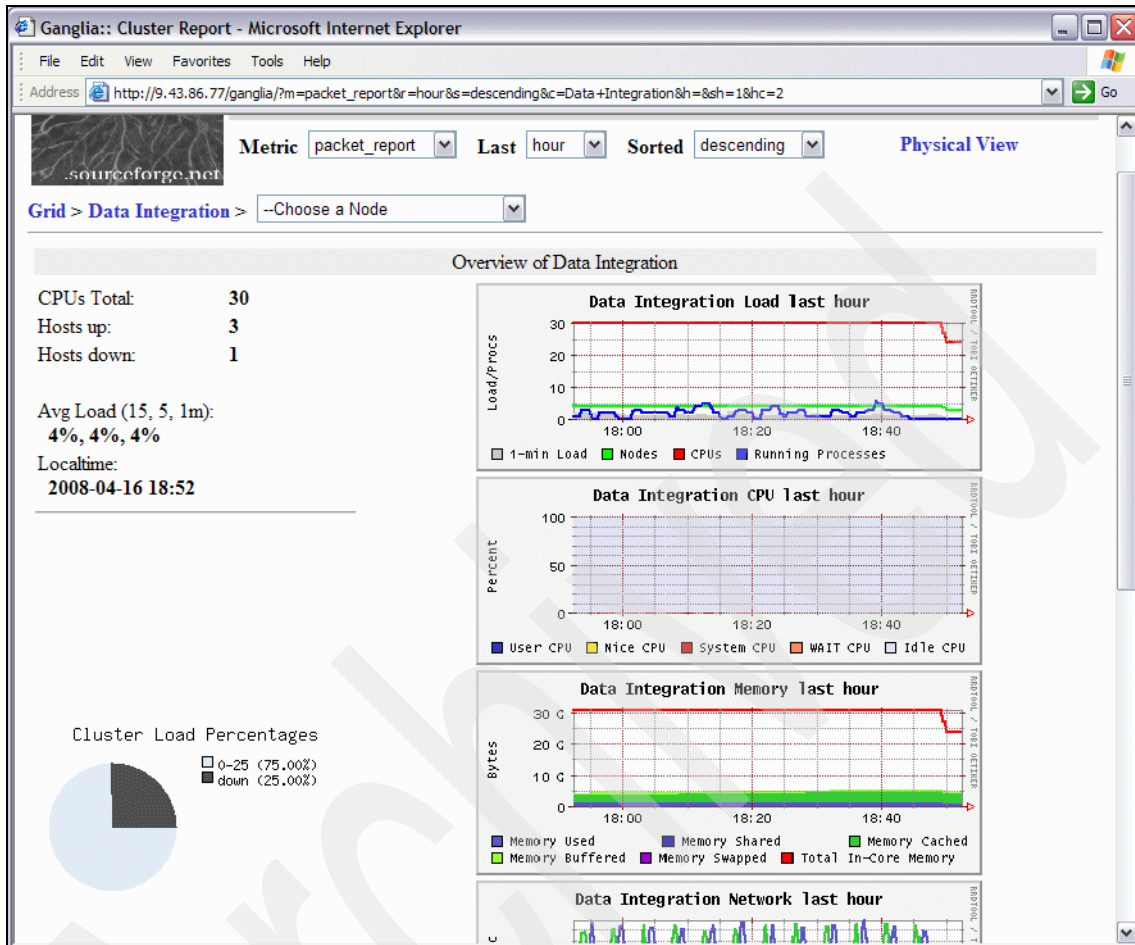


Figure 2-78 Overview of Data Integration, part 1 of 2

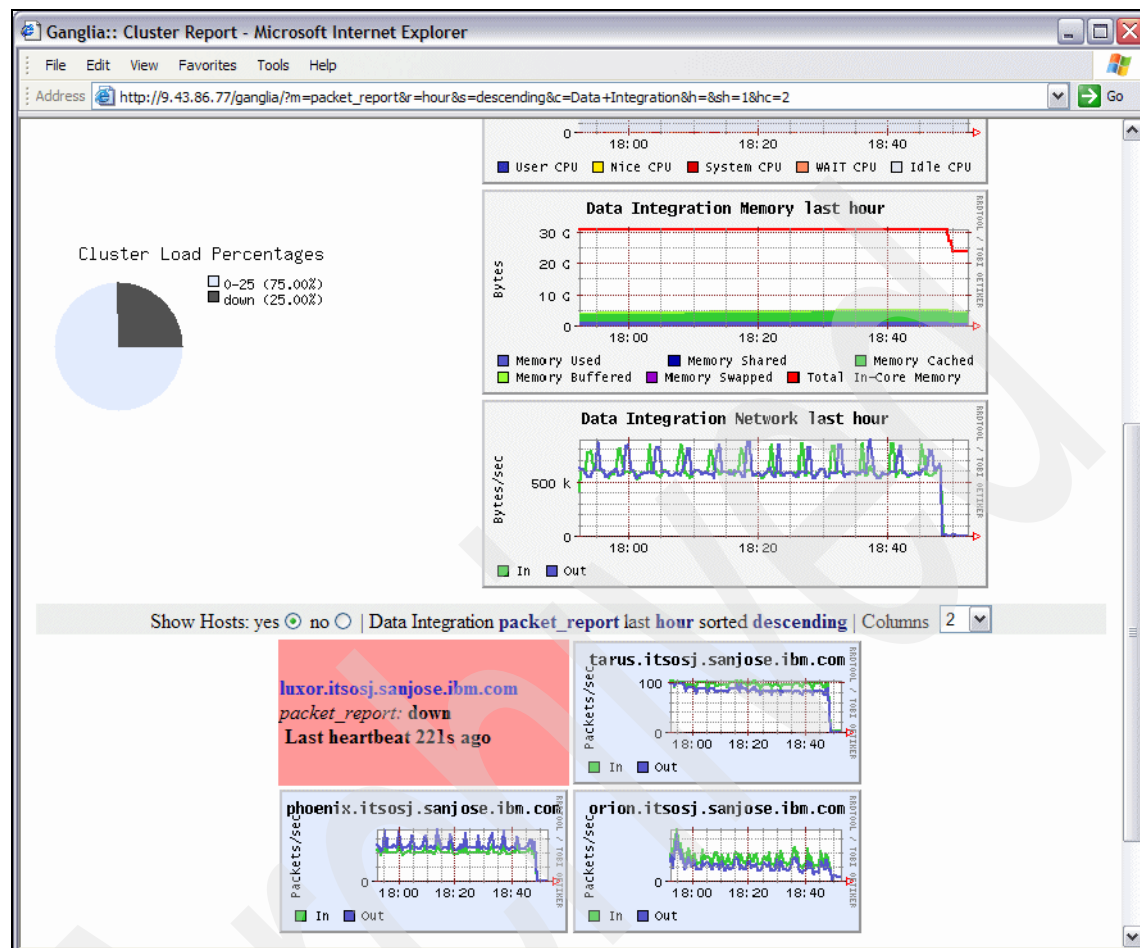


Figure 2-79 Overview of Data Integration, part 2 of 2

Figure 2-80 and Figure 2-81 on page 210 show the various levels of detail of the physical view based on the “Verbosity level” selected.

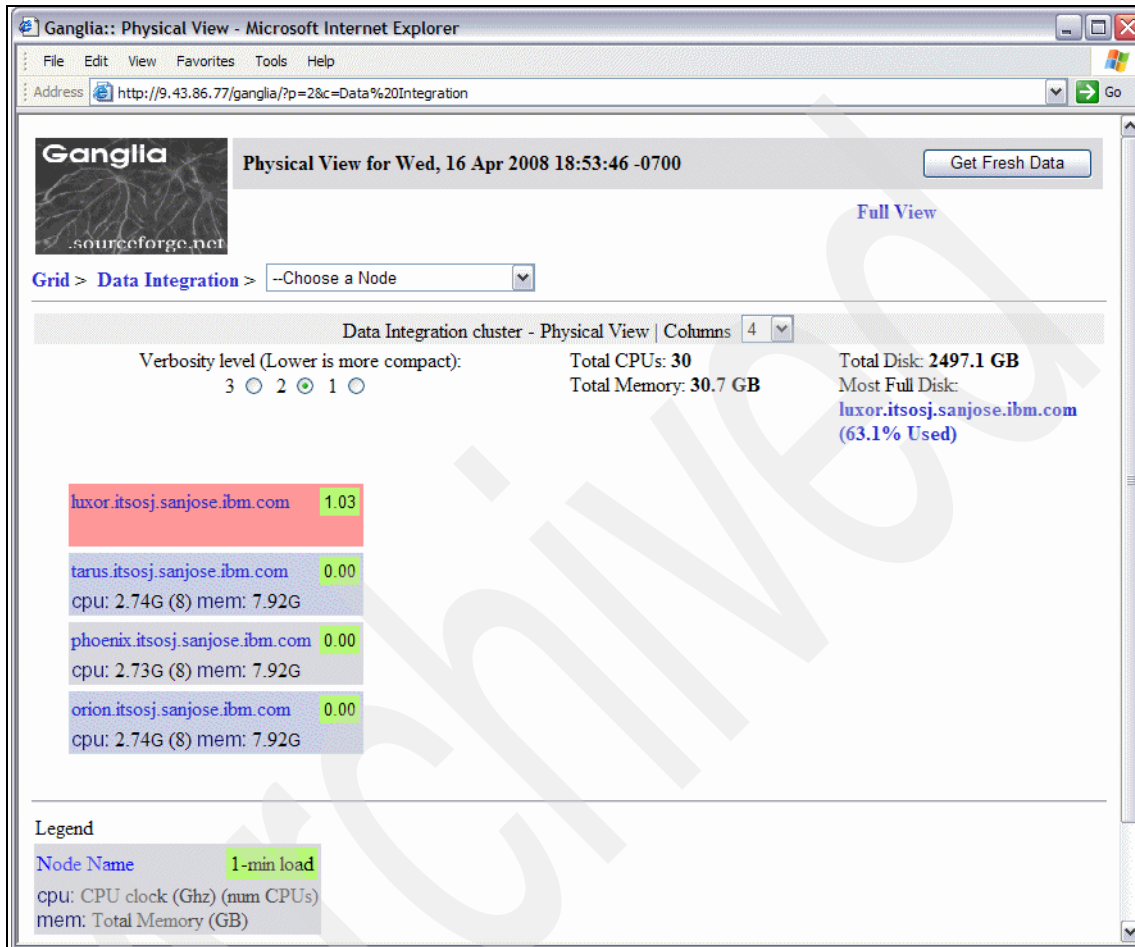


Figure 2-80 Physical view of nodes at different verbosity levels, part 1 of 2

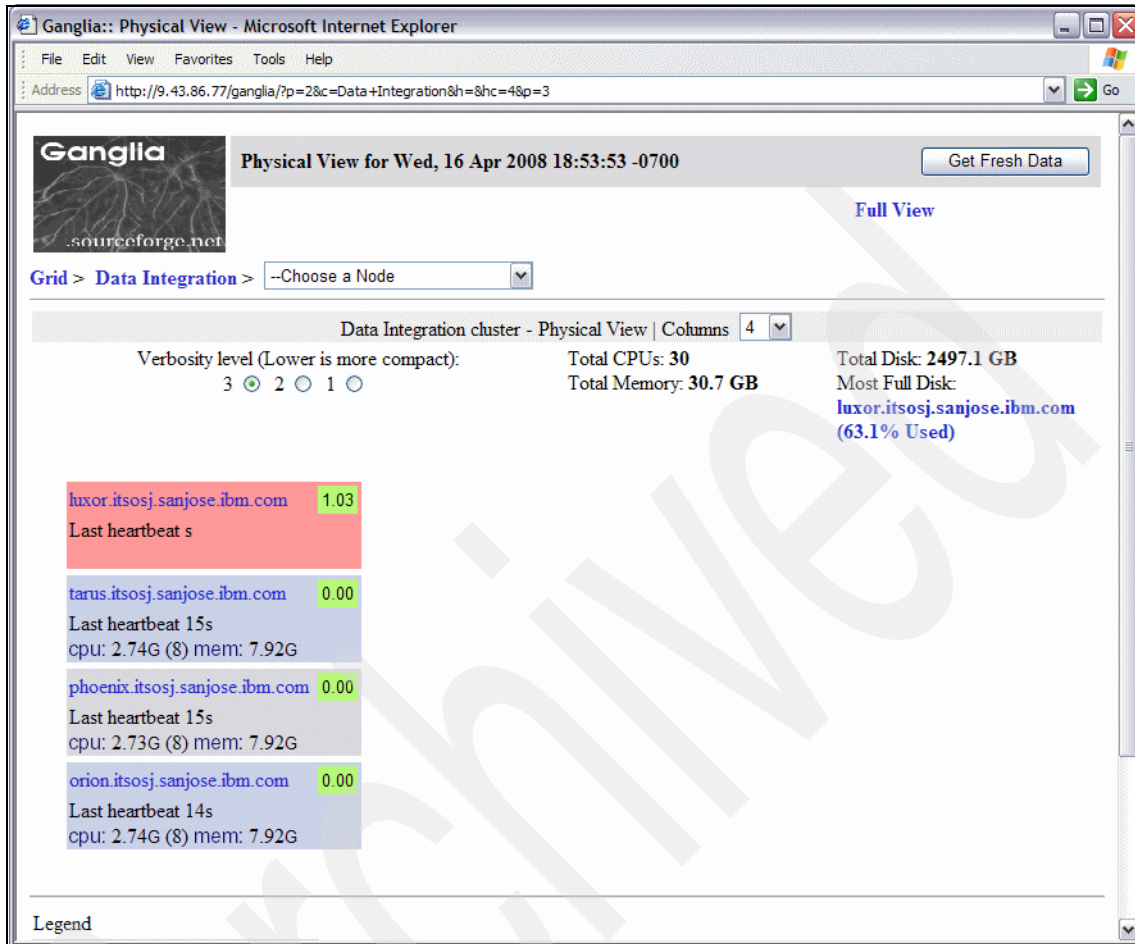


Figure 2-81 Physical view of nodes at different verbosity levels, part 2 of 2

Figure 2-82 shows detail of the node view of luxor.itsosj.sanjose.ibm.com

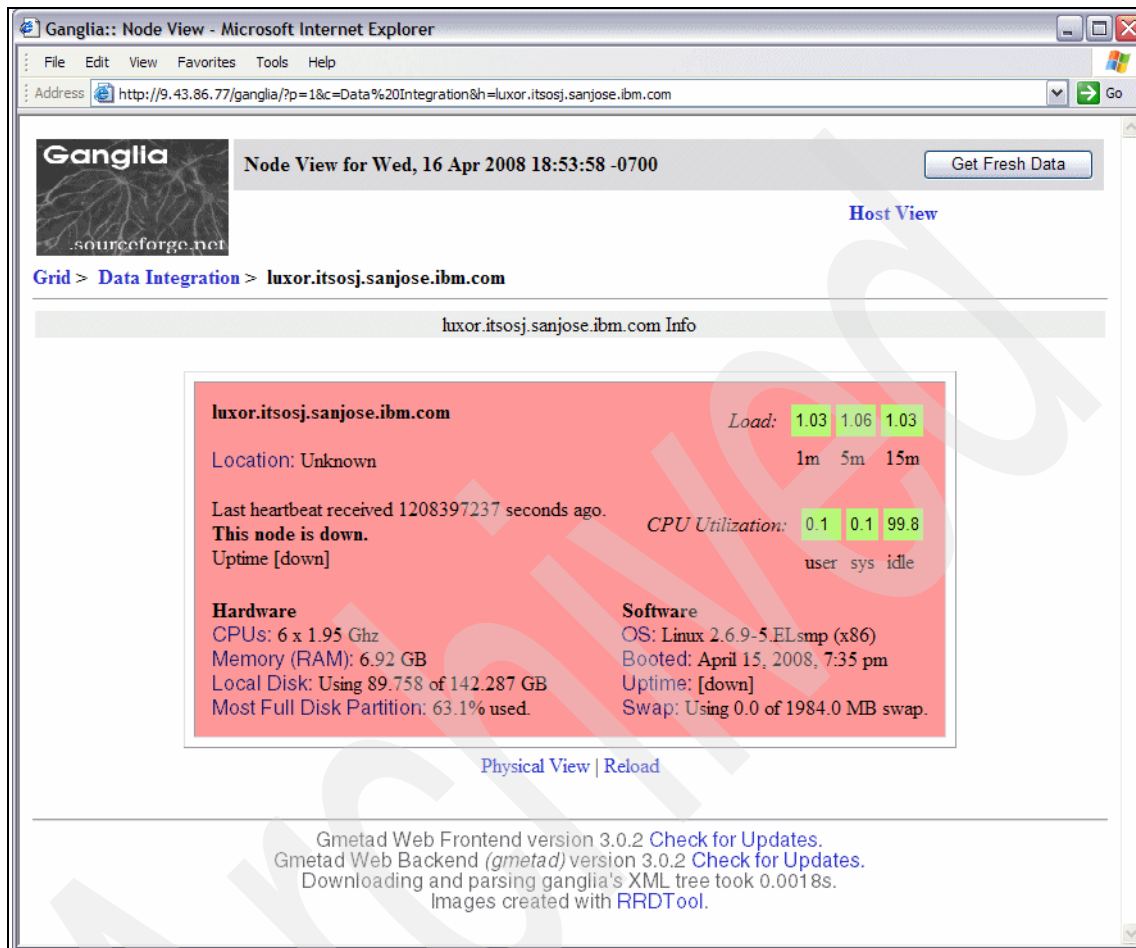


Figure 2-82 Node view of luxor.itsosj.sanjose.ibm.com

Figure 2-83 through Figure 2-85 on page 214 show the use of the individual compute nodes with the Metric (cpu_system) for the Last (hour).

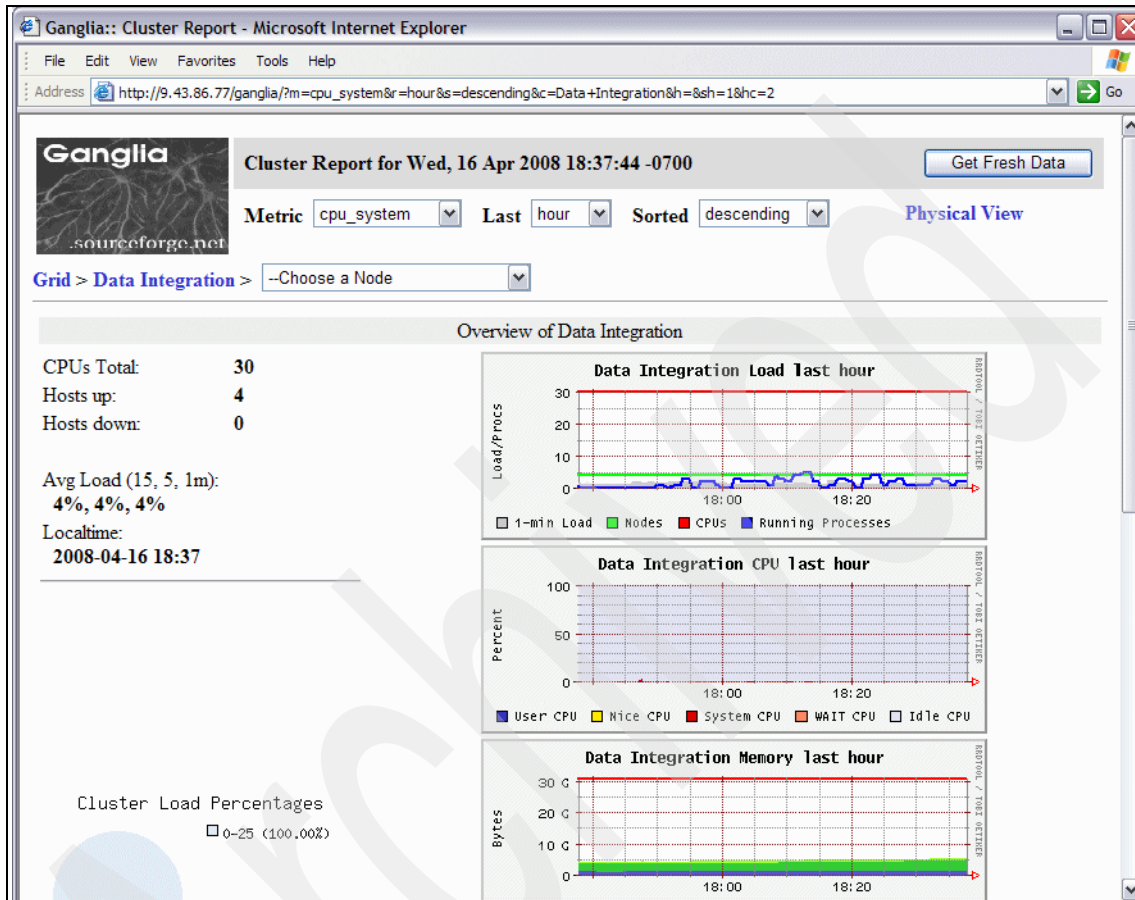


Figure 2-83 Use of the individual compute nodes, part 1 of 3

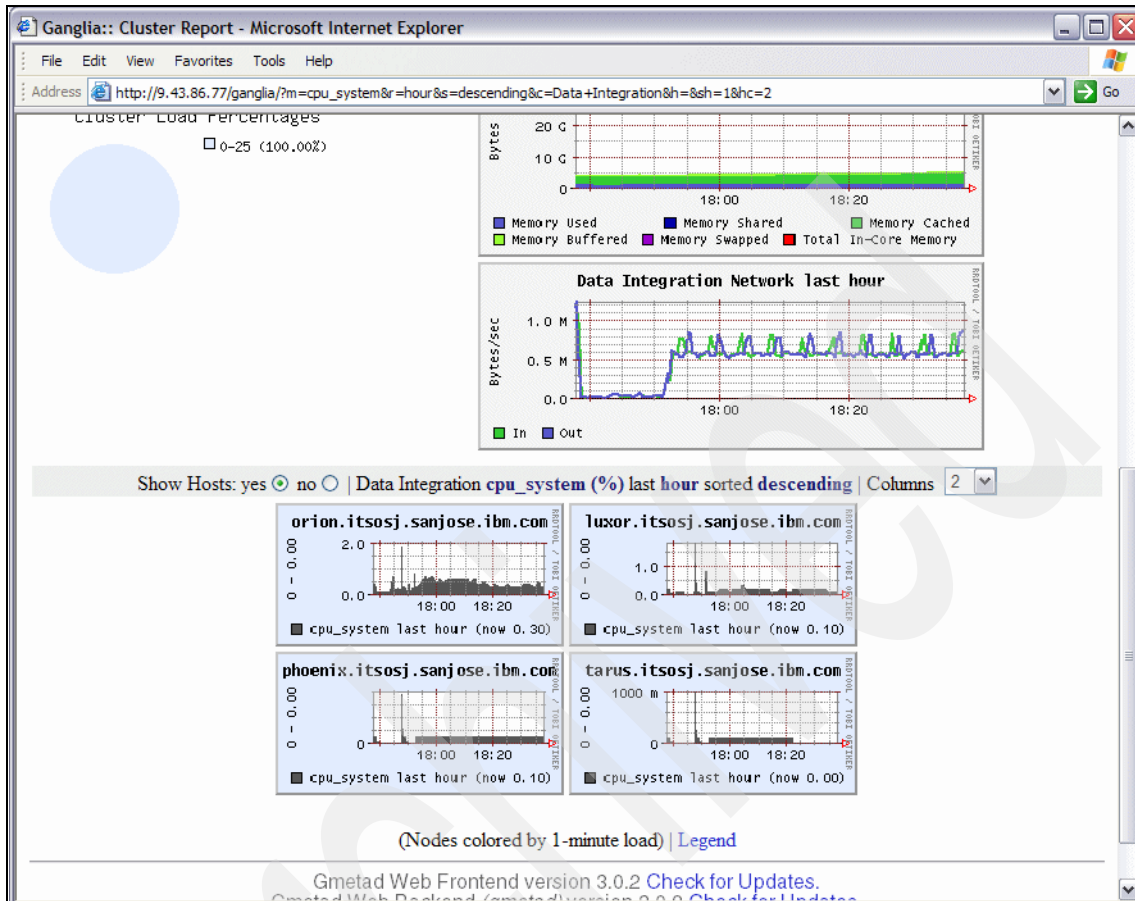


Figure 2-84 Use of the individual compute nodes, part 2 of 3

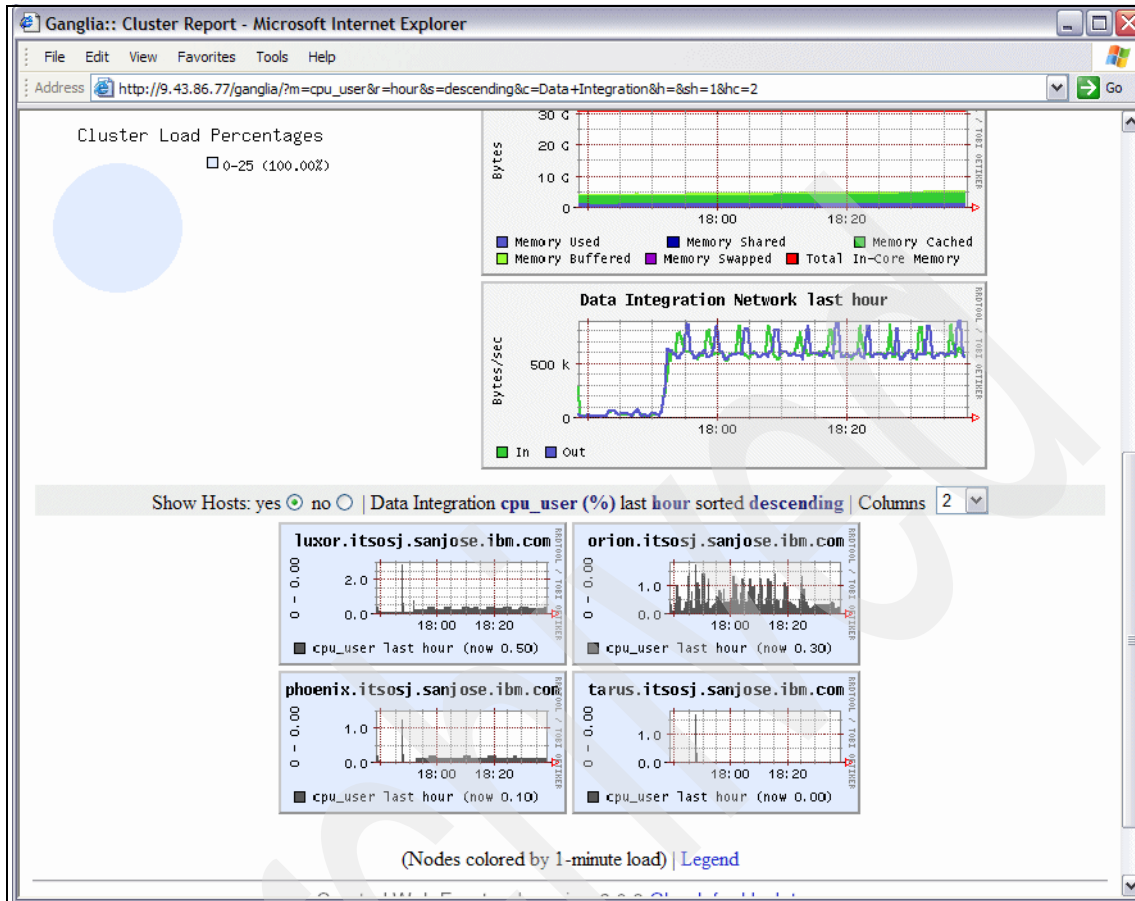


Figure 2-85 Use of the individual compute nodes, part 3 of 3

2.7.5 Cancelling a waiting job from the queue

If a job has been submitted into the LoadLeveler queue and resources are not available to release the job for execution, it remains in the queue until the required resources become available because there is no time-out mechanism.

If you want to remove the job from the queue, it must be cancelled as follows:

1. Identify the waiting jobs in the queue using the `lsthq` command as shown in Example 2-132 on page 215. The Step ID of the waiting job is orion.85.0.

Example 2-132 Cancel a waiting job, part 1 of 3

```
[dsadm@orion ~]$ listq
```

Step Id	Job Name	ST	PRI	Class	Running
On					

orion.85.0	rowGenSeqFile_28910	I	50	No_Class	
orion.84.0	rowGenSeq2Files_g_27	R	50	No_Class	tarus

2 job step(s) in queue, 1 waiting, 0 pending, 1 running, 0 held, 0 preempted

2. Issue the **canjob orion.85.0** command as shown in Example 2-132 and respond to the prompt with yes.

Example 2-133 Cancel a waiting job, part 2 of 3

```
[dsadm@orion grid_jobdir]$ canjob orion.85.0
orion.85.0          rowGenSeqFile_28910
cancel job (orion.85.0)? yes
llcancel: Cancel command has been sent to the central manager.
```

3. Issue the **listq** command again to verify that the waiting job is no longer in the queue as shown in Example 2-133.

Example 2-134 Cancel a waiting job, part 3 of 3

```
[dsadm@orion grid_jobdir]$ listq
```

Step Id	Job Name	ST	PRI	Class	Running
On					

orion.84.0	rowGenSeq2Files_g_27	R	50	No_Class	tarus

1 job step(s) in queue, 0 waiting, 0 pending, 1 running, 0 held, 0 preempted

The Director log output relating to the cancelled job orion.85.0 (Example 2-134) shows the Job cancelled message.

Server:KAZAN.ITSOSJ.SANJOSE.IBM.COM
Project:IS_Grid
Job No:7
Job name:rowGenSeqFile
Invocation:
Event Number:148
Event type:Info
User:dsadm
Timestamp:16/04/2008 6:10:54 PM
Message Id:XXA1805
Message:
<Dynamic_grid.sh> Waiting for Release from Queue

Server:KAZAN.ITSOSJ.SANJOSE.IBM.COM
Project:IS_Grid
Job No:7
Job name:rowGenSeqFile
Invocation:
Event Number:149
Event type:Info
User:dsadm
Timestamp:16/04/2008 6:12:39 PM
Message Id:XXA1810
Message:
<Dynamic_grid.sh> Waiting for Release from Queue
<Dynamic_grid.sh> Job cancelled.

Server:KAZAN.ITSOSJ.SANJOSE.IBM.COM
Project:IS_Grid
Job No:7
Job name:rowGenSeqFile
Invocation:
Event Number:150
Event type:Fatal
User:dsadm
Timestamp:16/04/2008 6:12:39 PM
Message Id:DSTAGE_RUN_E_0372
Message:
Parallel job reports failure (code 1)

2.7.6 Adding and removing nodes

In this section we describe how nodes can be added and removed and the impact it has on a job. We review the status of a job prior to nodes being removed, adding and removing a node, and the status of the same job after this change. Example 2-136 shows the output of two commands **llstatus** and **listq** while the job rowGenSeq2Files_g (which was defined to run with two compute nodes) was running.

- The **llstatus** command output shows a configuration with three nodes: One front end node (orion) and two compute nodes (phoenix and tarus). It also shows the two compute nodes running a job.
- The **listq** command output shows the job rowGenSeq2Files_g running in the No_Class queue.

Example 2-136 Phoenix and Tarus only (llstatus and listq)

```
[load1@orion ~]$ llstatus
Name                               Schedd  InQ Act Startd Run LdAvg Idle Arch
OpSys
orion.itsosj.sanjose.ibm. Avail      0  0 None    0 0.04  119 i386
Linux2
phoenix.itsosj.sanjose.ib Avail      0  0 Busy    1 0.00  6337 i386
Linux2
tarus.itsosj.sanjose.ibm. Avail      0  0 Busy    1 0.05  6738 i386
Linux2

i386/Linux2                3 machines      0 jobs      2 running
Total Machines              3 machines      0 jobs      2 running

The Central Manager is defined on orion.itsosj.sanjose.ibm.com

The BACKFILL scheduler is in use

All machines on the machine_list are present.

[load1@orion ~]$ listq
Step Id                        Job Name                ST PRI Class      Running
On
-----
-----
orion.86.0                    rowGenSeq2Files_g_74 R  50 No_Class  tarus

1 job step(s) in queue, 0 waiting, 0 pending, 1 running, 0 held, 0
preempted
```

Example 2-137 shows the content of the LoadL_admin file on the front end node before adding another node, the content of the LoadL_admin file on the front end node after adding removing the phoenix node and adding node luxor, and the output of the **llstatus** command.

- ▶ The LoadL_admin file contents before adding the luxor node included orion (as the front end node) and the two compute nodes tarus and phoenix.
- ▶ The LoadL_admin file contents after adding the luxor node and removing the phoenix node included orion (as the front end node) and the two compute nodes luxor and tarus.

Note: For these changes to take effect, you must issue the **llctl reconfig** command.

After the changes have taken effect, the **llstatus** command output shows a configuration with three nodes: One front end node (orion) and two compute nodes (luxor and tarus). It also shows the two compute nodes being busy running a job.

Example 2-137 Add Luxor and drop Phoenix

1) LoadL_admin file BEFORE adding Luxor

```
...
###
orion:      type = machine
            # adapter_stanzas = orionpvt.itsosj.sanjose.ibm.com
            # alias = luxor
            central_manager = true
            schedd_host = true
tarus:      type = machine
            # adapter_stanzas = taruspvt.itsosj.sanjose.ibm.com
            # alias = tarus
            central_manager = false
            schedd_host = false
phoenix:    type = machine
            # adapter_stanzas = phoenixpvt.itsosj.sanjose.ibm.com
            # alias = phoenix
            central_manager = false
            schedd_host = false
```

2) LoadL_admin file AFTER removing Phoenix and adding Luxor

```
...
###
orion:      type = machine
```



```

# adapter_stanzas = orionpvt.itsosj.sanjose.ibm.com
# alias = luxor
central_manager = true
schedd_host = true
tarus: type = machine
# adapter_stanzas = taruspvt.itsosj.sanjose.ibm.com
# alias = tarus
central_manager = false
schedd_host = false
luxor: type = machine
# adapter_stanzas = luxorpvt.itsosj.sanjose.ibm.com
# alias = tarus
central_manager = false
schedd_host = false

```

3) tell Loadleveler that the Load_admin file has been changed:

```

[loadl@orion ~]$ llctl reconfig
llctl: Sent reconfig command to host orion.itsosj.sanjose.ibm.com

```

4) View LoadLeveler status

```

[loadl@orion ~]$ llstatus

```

Name	Schedd	InQ	Act	Startd	Run	LdAvg	Idle	Arch
OpSys								
luxor.itsosj.sanjose.ibm.	Avail	0	0	Idle	0	1.00	5	i386
Linux2								
orion.itsosj.sanjose.ibm.	Avail	0	0	None	0	0.00	0	i386
Linux2								
tarus.itsosj.sanjose.ibm.	Avail	0	0	Idle	0	0.00	7099	i386
Linux2								
i386/Linux2	3 machines			0 jobs		0	running	
Total Machines	3 machines			0 jobs		0	running	

The **Central Manager** is defined on **orion.itsosj.sanjose.ibm.com**

The BACKFILL scheduler is in use

All machines on the machine_list are present.

After the compute node changes, Example 2-138 on page 220 shows the output of two commands **llstatus** and **llstq** while the job rowGenSeq2Files_g (which was defined to run with two compute nodes) was running.

The **llstatus** command output shows a configuration with three nodes: One front end node (orion) and two compute nodes (luxor and tarus). It also shows the two compute nodes being busy running a job.

The **listq** command output shows the job rowGenSeq2Files_g running in the No_Class queue.

Attention: This scenario highlights the dynamic aspect of the grid environment where the addition or removal of compute nodes is handled without having to modify the configuration files used by the jobs.

Example 2-138 Tarus and Luxor only (llstatus and listq)

[load1@orion ~]\$ llstatus

Name	Schedd	InQ	Act	Startd	Run	LdAvg	Idle	Arch
OpSys								
luxor.itsosj.sanjose.ibm. Linux2	Avail	0	0	Busy	1	0.00	6932	i386
orion.itsosj.sanjose.ibm. Linux2	Avail	0	0	None	0	0.00	83	i386
tarus.itsosj.sanjose.ibm. Linux2	Avail	0	0	Busy	1	0.00	7333	i386
i386/Linux2	3 machines			0 jobs	2	running		
Total Machines	3 machines			0 jobs	2	running		

The Central Manager is defined on orion.itsosj.sanjose.ibm.com

The BACKFILL scheduler is in use

All machines on the machine_list are present.

[load1@orion ~]\$ listq

Step Id	Job Name	ST	PRI	Class	Running
On					

orion.88.0	rowGenSeq2Files_g_88	R	50	No_Class	tarus

1 job step(s) in queue, 0 waiting, 0 pending, **1 running**, 0 held, 0 preempted

2.8 Step 4: Failing over to the standby node NILE

As part of our HA solution, we have a dedicated standby node (nile.itsosj.sanjose.ibm.com). When the front end (orion.itsosj.sanjose.ibm.com) fails, a takeover procedure must transfer the front-end role from orion to nile which was configured to adopt this role in 2.5.7, “Step 1g: Configuring the standby node” on page 143.

At takeover, the standby node must have access to all the components of the primary front end node including IBM Information Server, Workload Scheduler LoadLeveler, Ganglia, NFS, NIS, and the virtual IP address and corresponding domain name.

Note: Many of the following steps are required because our HA solution involved a manual takeover. In an automated HA solution, steps such as determining whether the services must be moved from the primary front end node to the standby node, and that remounts of shared devices on a SAN are not required.

The takeover procedure includes the following steps:

1. Ensure that the primary front end node (orion) is down. Specifically the virtual IPs. You can do this by pinging the virtual IP address (kazan.itsosj.sanjose.ibm.com) as shown in Example 2-139.

Example 2-139 ping kazan.itsosj.sanjose.ibm.com

```
[root@nile ~]# ping -c 1 kazan
PING kazan.itsosj.sanjose.ibm.com (9.43.86.77) 56(84) bytes of data.
From nile.itsosj.sanjose.ibm.com (9.43.86.76) icmp_seq=0 Destination
Host Unreachable

--- kazan.itsosj.sanjose.ibm.com ping statistics ---
1 packets transmitted, 0 received, +1 errors, 100% packet loss, time
0ms
, pipe 2
```

Note: If the primary front end node is still up with its virtual IP address, there will be a TCP/IP address conflict when the virtual IP is brought up on the standby node (nile). Therefore, ensure that the IP addresses are not available by using the ping command.

After the failure of the front end node (orion), a **df** command issued from one of the compute nodes (tarus) hangs as shown in Example 2-140.

Example 2-140 df command output on one of the compute nodes after front end node failure (hangs)

[root@tarus ~]# df					
Filesystem	1K-blocks	Used	Available	Use%	Mounted on
/dev/mapper/VolGroup00-LogVol100					
	574536368	2742948	542608604	1%	/
/dev/sda1	101086	13643	82224	15%	/boot
none	4154816	0	4154816	0%	/dev/shm

2. Activate the virtual IP addresses by issuing the **ifup** command on the eth0:1 and eth1:1 interfaces.

After the activation, a **df** command issued from one of the compute nodes (tarus) shows the command output on the tarus compute nodes (Example 2-141).

Example 2-141 df command output on one of the compute nodes (tarus)

[root@tarus ~]# df					
Filesystem	1K-blocks	Used	Available	Use%	Mounted on
/dev/mapper/VolGroup00-LogVol100					
	574536368	2742948	542608604	1%	/
/dev/sda1	101086	13643	82224	15%	/boot
none	4154816	0	4154816	0%	/dev/shm
192.168.1.77:/home	-	-	-	-	/home
192.168.1.77:/ds_overview	-	-	-	-	/ds_overview
192.168.1.77:/opt/IBM	-	-	-	-	/opt/IBM

3. Mount the following file systems:

- /opt/IBM
- /home
- /ds_overview
- /opt/mqm

Note: You must ensure that the device names are identical on the frontend and the standby node, as this affects the mount command. If the device names are different (as in our case), because of differences in the disk configuration, you cannot use the same mount command on the standby node for the file systems that was used on the front end node.

The device names and mounted file systems can be determined by issuing the **fdisk -l** and **df** commands on the frontend and standby node as shown in Example 2-142 and Example 2-143 respectively.

The **fdisk -l** command output on the standby node is shown in Example 2-144 on page 224.

The device names corresponding to the shared SAN volume is sdf[n] on the front end node (orion) and sdd[n] on the standby node (nile).

Example 2-142 fdisk -l command on the front end node (orion)

```
[root@orion ~]# fdisk -l
```

Disk /dev/sda: 300.0 GB, 300000000000 bytes
 255 heads, 63 sectors/track, 36472 cylinders
 Units = cylinders of 16065 * 512 = 8225280 bytes

Device	Boot	Start	End	Blocks	Id	System
--------	------	-------	-----	--------	----	--------

Disk /dev/sdf: 598.9 GB, 598925639680 bytes
 255 heads, 63 sectors/track, 72815 cylinders
 Units = cylinders of 16065 * 512 = 8225280 bytes

Device	Boot	Start	End	Blocks	Id	System
/dev/sdf1		1	24900	200009218+	83	Linux
/dev/sdf2		24901	37350	100004625	83	Linux
/dev/sdf3		37351	49800	100004625	83	Linux
/dev/sdf4		49801	72815	184867987+	5	Extended
/dev/sdf5		49801	58800	72292468+	83	Linux

Example 2-143 df command on the front end node (orion)

```
[root@orion ~]# df
```

Filesystem	1K-blocks	Used	Available	Use%	Mounted on
/dev/mapper/VolGroup00-LogVol100	574536368	49687220	495664332	10%	/
/dev/sda1	101086	15340	80527	17%	/boot
none	4154816	0	4154816	0%	/dev/shm
/dev/sdf1	196870980	32482408	154388112	18%	/home
/dev/sdf2	98435476	13792436	79642812	15%	/opt/IBM
/dev/sdf3	98435476	95844	93339404	1%	/ds_overview
/dev/sdf5	71157244	554568	66988056	1%	/opt/mqm

```
[root@nile ~]# fdisk -l
```

```
Disk /dev/sda: 300.0 GB, 300000000000 bytes
255 heads, 63 sectors/track, 36472 cylinders
Units = cylinders of 16065 * 512 = 8225280 bytes
```

```
.....
```

```
Disk /dev/sdd: 598.9 GB, 598925639680 bytes
255 heads, 63 sectors/track, 72815 cylinders
Units = cylinders of 16065 * 512 = 8225280 bytes
```

Device	Boot	Start	End	Blocks	Id	System
/dev/sdd1		1	24900	200009218+	83	Linux
/dev/sdd2		24901	37350	100004625	83	Linux
/dev/sdd3		37351	49800	100004625	83	Linux
/dev/sdd4		49801	72815	184867987+	5	Extended
/dev/sdd5		49801	58800	72292468+	83	Linux

4. Change the physical host name of the front end node of the LoadL_admin configuration file from ORION to NILE.

Note: Once the standby node takes over the front-end role, the LoadL service must be reloaded by issuing the `llctl reconfig` command on the front end node running as the loadl user.

You do not need to change the LoadL_config.local file because it must be an exact copy of the one that is running on the front end node. Because this is a shared file, it requires no change.

5. Start NFS services.
6. Start Ganglia services gmond, gmetad, and httpd.
7. Start IBM Information Server.
8. Start LoadLeveler.
9. Remount the file systems on the compute nodes.

Note: A number of scripts are provided with the installation of IBM Information Server. They can be found in the /etc/rc3.d or /etc/init.d directories. They can be used without modification.

The full set of scripts include: ganglia (gmond gmetad), LoadLeveler (LoadL), ISFAgents, ISFServer, and ds.rc (search for full name in /etc/init.d and /etc/rc3.d).

These scripts must be moved to a location outside of the normal startup directory (rc?.d or init.d). After you move these scripts to a common directory that is shared between both head nodes, you must develop a script to start or stop each service in the correct order. The start order is listed above, while the stop order is the reverse of it.

Note: DB2 does not have a start/stop script, but must be removed from the /etc/inittab file (last line in the file) to prevent auto startup when the system boots. You can start db2 with the following command:

```
su -lc "db2start" db2inst1
```

You can stop db2 with the following command:

```
su -lc "db2stop" db2inst1
```

This command must be the first script started and the last script stopped.

In our case, we developed the following scripts to simplify the takeover process from the front end node orion to the standby node nile:

- ▶ The failover_Nile.sh script (shown in Example B-23 on page 289) executes on the standby node (nile) and performs the following tasks identified earlier:
 - a. Ensures that the primary front end node (orion) is down
 - b. Activates the virtual IP addresses
 - c. Mounts the file systems /opt/IBM, /home, /ds_overview and /opt/mqm
 - d. Changes the physical host name of the front end node of the LoadL_admin configuration file from ORION to NILE
 - e. Starts NFS services
 - f. Starts Ganglia services gmond, gmetad, and httpd.
 - g. Starts IBM Information Server.
 - h. Starts LoadLeveler.

Example 2-145 on page 226 shows the output of the execution of this script.

Example 2-145 failover_Nile.sh script output

```
[root@nile ~]# ./failover_Nile.sh
Checking if virtual IP addresses are still up...(this takes
approximately 3
seconds.)
Setting current host as primary headnode...
Upping Virtual IP Addresses
eth0:1    Link encap:Ethernet HWaddr 00:02:55:AC:86:A4
          inet addr:9.43.86.77 Bcast:9.43.87.255 Mask:255.255.252.0
          UP BROADCAST RUNNING MULTICAST MTU:1500 Metric:1

eth1:1    Link encap:Ethernet HWaddr 00:02:55:E4:5B:17
          inet addr:192.168.1.77 Bcast:192.168.3.255 Mask:255.255.252.0
          UP BROADCAST RUNNING MULTICAST MTU:1500 Metric:1

Mounting SAN filesystems
Filesystem      1K-blocks      Used Available Use% Mounted on
/dev/mapper/VolGroup00-LogVol100
574278336 97647112 447459512 18% /
/dev/sda1      101086      15028      80839 16% /boot
none          3628496      0      3628496 0% /dev/shm
/dev/sdd1      196870980 32220088 154650432 18% /home
/dev/sdd2      98435476 13108476 80326772 15% /opt/IBM
/dev/sdd3      98435476      95748 93339500 1% /ds_overview
/dev/sdd5      71157244 554568 66988056 1% /opt/mqm
Setting LoadL_admin configuration file central_manager to nile
Restarting NFS server
Shutting down NFS mountd:
[FAILED]
Shutting down NFS daemon:
[FAILED]
Shutting down NFS quotas:
[FAILED]
Shutting down NFS services:
[ OK ]
Starting NFS services:
[ OK ]
Starting NFS quotas:
[ OK ]
```

- The remount_Nile.sh script (shown in Example B-25 on page 293) executes on the standby node (nile) and mounts the file systems on the compute nodes.

The nodes.cfg file contains a list of all the compute nodes as shown in Example 2-146 on page 227. This file is referenced in the script to determine all the compute nodes which require file systems mounted.


```
tarus
phoenix
luxor
```

Example 2-147 shows the output of the execution of this script.

Note: We found that some remnant processes can prevent the mounting of file systems on the compute nodes. Therefore, we first had to kill such processes before performing the mount of the file systems.

```
[root@nile ~]# ./remount_Nile.sh
tarus
phoenix
luxor
-----
kill resource tracker on the above compute nodes
<node name="tarus" command="kill -9 $(lsdf 2> /dev/null | grep
'/opt/IBM' | grep 'resour' | awk '{print $2}' | sort | uniq) ">
</node>
<node name="phoenix" command="kill -9 $(lsdf 2> /dev/null | grep
'/opt/IBM' | grep 'resour' | awk '{print $2}' | sort | uniq) ">
</node>
<node name="luxor" command="kill -9 $(lsdf 2> /dev/null | grep
'/opt/IBM' | grep 'resour' | awk '{print $2}' | sort | uniq) ">
</node>
-----
run umount -a
<node name="tarus" command="umount -a ">
umount: /var/lib/nfs/rpc_pipefs: device is busy
umount: /: device is busy
</node>
<node name="phoenix" command="umount -a ">
umount: /var/lib/nfs/rpc_pipefs: device is busy
umount: /: device is busy
</node>
<node name="luxor" command="umount -a ">
umount: /var/lib/nfs/rpc_pipefs: device is busy
umount: /: device is busy
</node>
-----
run df
```

```

<node name="tarus" command="df ">
Filesystem            1K-blocks      Used Available Use% Mounted on
/dev/mapper/VolGroup00-LogVol100
                        574536368    2743556 542607996   1% /
</node>
<node name="phoenix" command="df ">
Filesystem            1K-blocks      Used Available Use% Mounted on
/dev/mapper/VolGroup00-LogVol100
                        574536368    2732364 542619188   1% /
</node>
<node name="luxor" command="df ">
Filesystem            1K-blocks      Used Available Use% Mounted on
/dev/mapper/VolGroup00-LogVol100
                        138851328    80517340 51280676   62% /
</node>
-----
sleep for 3 minutes
-----
run mount -a
<node name="tarus" command="mount -a ">
</node>
<node name="phoenix" command="mount -a ">
</node>
<node name="luxor" command="mount -a ">
</node>
-----
run df
<node name="tarus" command="df ">
Filesystem            1K-blocks      Used Available Use% Mounted on
/dev/mapper/VolGroup00-LogVol100
                        574536368    2743564 542607988   1% /
/dev/sda1              101086      13643     82224   15% /boot
none                   4154816        0    4154816    0% /dev/shm
192.168.1.77:/home     196871008   32220768 154649792   18% /home
192.168.1.77:/ds_overview
                        98435488     95744   93339520   1% /ds_overview
192.168.1.77:/opt/IBM
                        98435488   13108640 80326592   15% /opt/IBM
</node>
<node name="phoenix" command="df ">
Filesystem            1K-blocks      Used Available Use% Mounted on
/dev/mapper/VolGroup00-LogVol100
                        574536368    2732368 542619184   1% /
/dev/sda1              101086      13644     82223   15% /boot
none                   4154816        0    4154816    0% /dev/shm

```

```

192.168.1.77:/home 196871008 32220768 154649792 18% /home
192.168.1.77:/ds_overview
98435488 95744 93339520 1% /ds_overview
192.168.1.77:/opt/IBM
98435488 13108640 80326592 15% /opt/IBM
</node>
<node name="luxor" command="df ">
Filesystem 1K-blocks Used Available Use% Mounted on
/dev/mapper/VolGroup00-LogVol100
138851328 80517340 51280676 62% /
/dev/sda1 101086 12393 83474 13% /boot
none 3628544 0 3628544 0% /dev/shm
192.168.1.77:/home 196871008 32220768 154649792 18% /home
192.168.1.77:/ds_overview
98435488 95744 93339520 1% /ds_overview
192.168.1.77:/opt/IBM
98435488 13108640 80326592 15% /opt/IBM
</node>

```

We ran the rowGenSleeper_g job described in “Single jobs” on page 148 (with three compute nodes instead of two, and two partitions instead of 1) to show that the standby node (nile) was performing the front end node role successfully as follows:

1. Click the icon to run the job in Figure 2-86 on page 230 which brings up the Job Run Options window.
2. Modify the following parameter values in the Job Run Options window as shown in Figure 2-87 on page 231:
 - ENABLE - Use DYNAMIC_CONFIG_SCRIPT (NO/YES/OLD) value to YES
 - COMPUTENODES - Number of Compute Nodes value to 3
 - PARTITIONS - Number of partitions per Node (compute) value to two

Figure 2-88 on page 231 shows the successful execution of the job where 10 rows are generated and the same number are written to the Peek stage.

The output of the DataStage and QualityStage Director log (Figure 2-89 on page 232) shows the servers luxorpvt.itsosj.sanjose.ibm.com, phoenixpvt.itsosj.sanjose.ibm.com and taruspvt.itsosj.sanjose.ibm.com as the compute nodes, and the configuration file used as being rowGenSleeper_g_31581.config.apt.

The Event Detail log (Figure 2-90 on page 232) indicates the parameter values used in the execution of this job.

The Event Detail log (Figure 2-91 on page 233) indicates the submission of the job to the front end node `nile.itsosj.sanjose.ibm.com`.

The contents of the generated `rowGenSleeper_g_31581.config.apf` file (Figure 2-92 on page 233 and Figure 2-93 on page 234) has six nodes in all:

- Two (Compute1 and Compute2) corresponding to the `luxorpvt.itsosj.sanjose.ibm.com` machine.
- Two (Compute3 and Compute4) corresponding to the `phoenixpvt.itsosj.sanjose.ibm.com` machine.
- Two (Compute5 and Compute6) corresponding to the `taruspvt.itsosj.sanjose.ibm.com` machine.
- The Conductor node is the `kazanpvt.itsosj.sanjose.ibm.com` machine.

Example 2-148 on page 235 shows the output of the `llmonitor` command while this job is running. It shows one job running on three nodes (Startd column shows Busy), and the Central Manager being defined on `nile.itsosj.sanjose.ibm.com`.

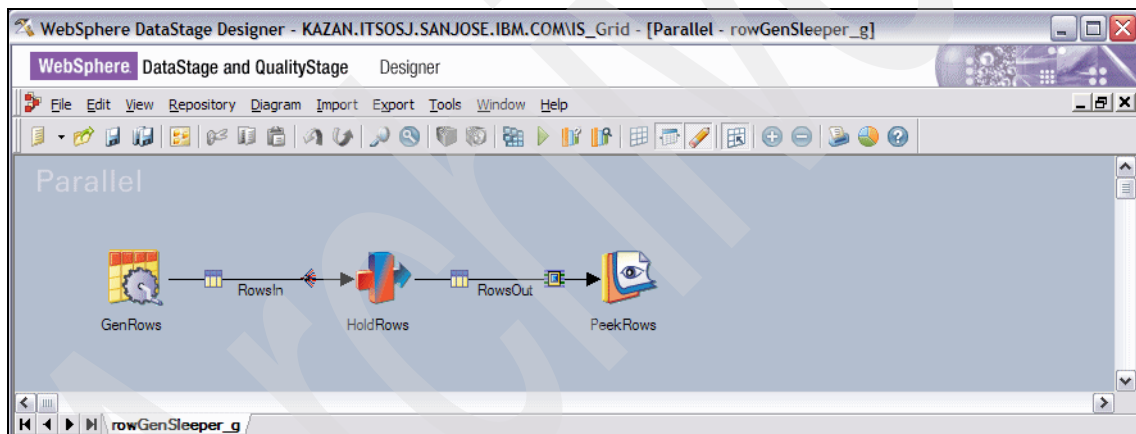


Figure 2-86 `rowGenSleeper_g` after failover, part 1 of 8

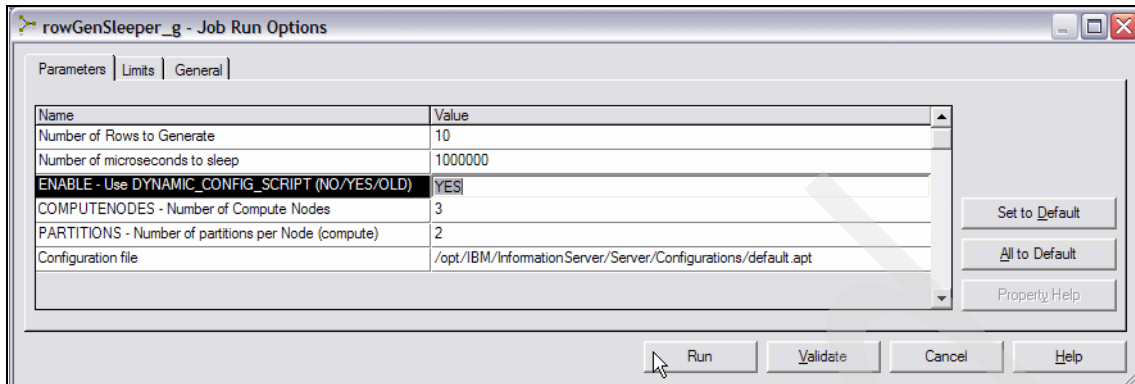


Figure 2-87 rowGenSleeper_g after failover, part 2 of 8

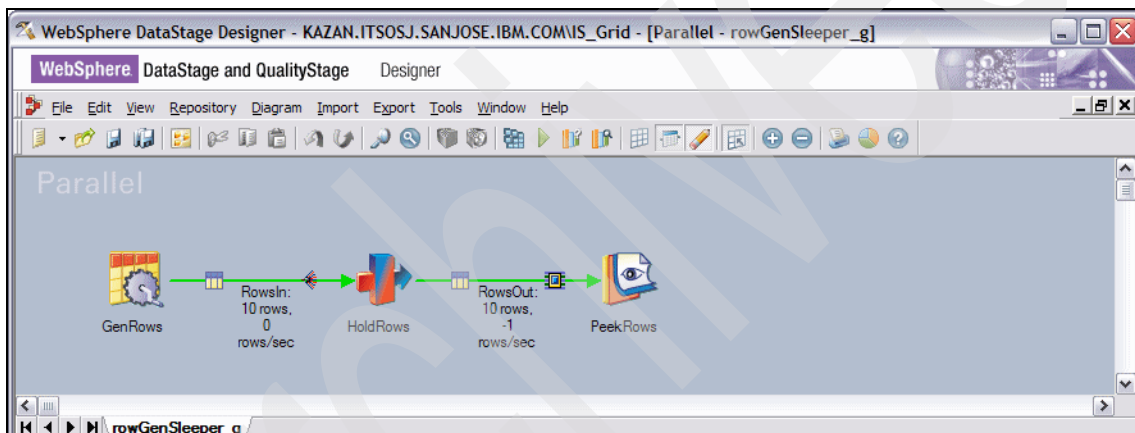


Figure 2-88 rowGenSleeper_g after failover, part 3 of 8

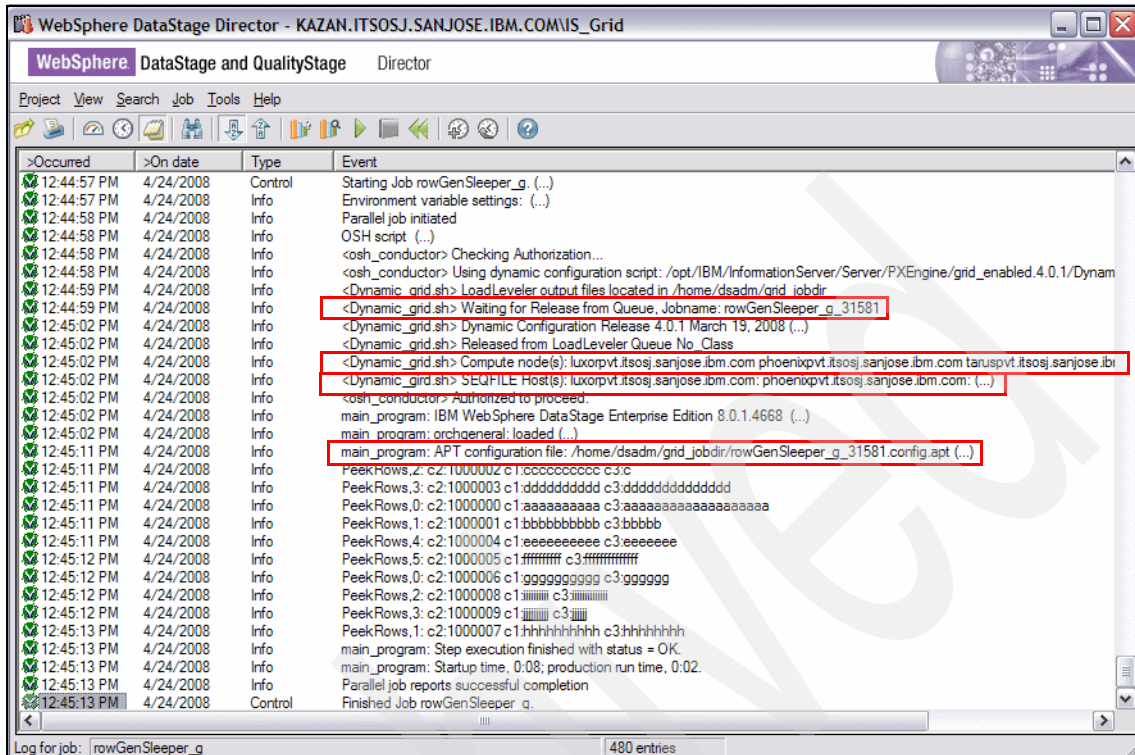


Figure 2-89 rowGenSleeper_g after failover, part 4 of 8

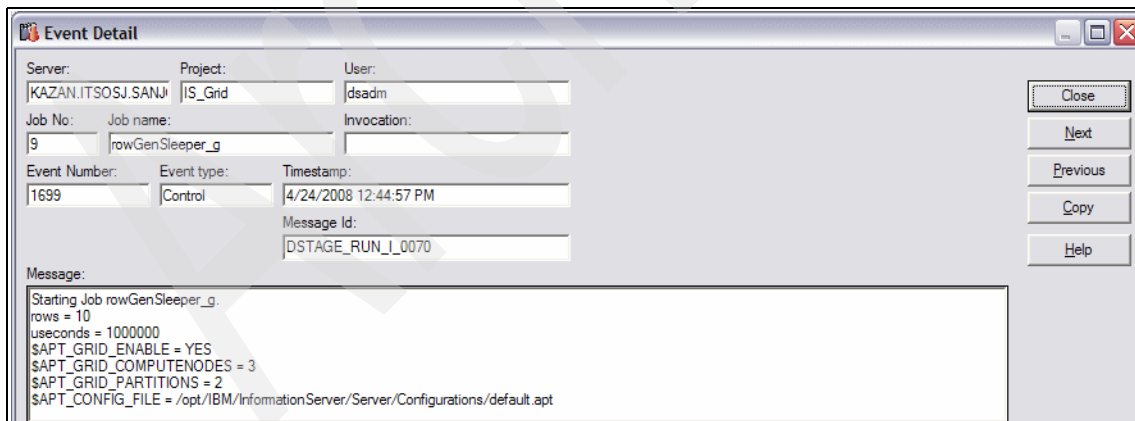


Figure 2-90 rowGenSleeper_g after failover, part 5 of 8

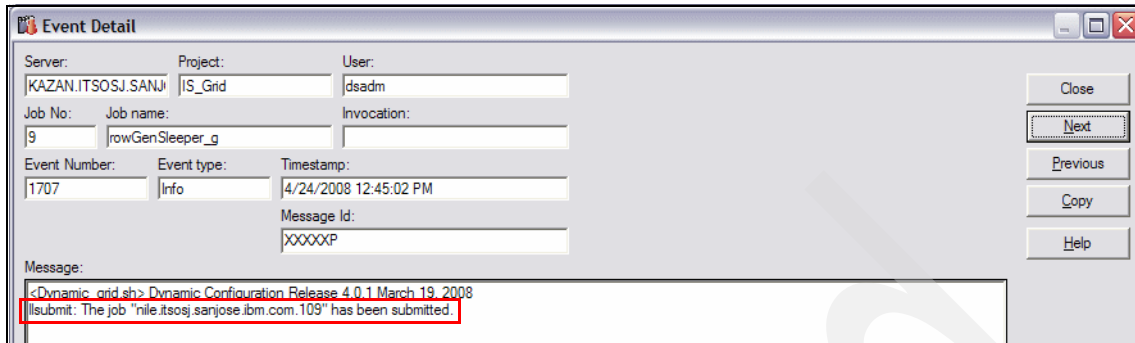


Figure 2-91 rowGenSleeper_g after failover, part 6 of 8

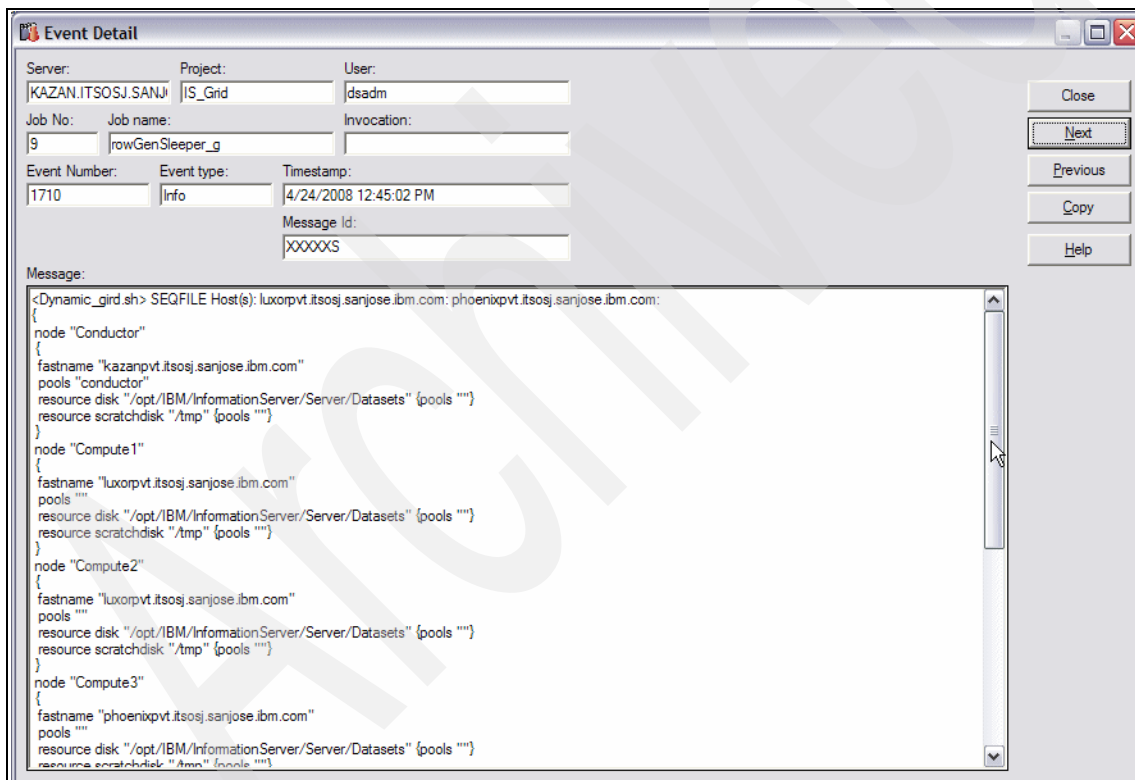


Figure 2-92 rowGenSleeper_g after failover, part 7 of 8

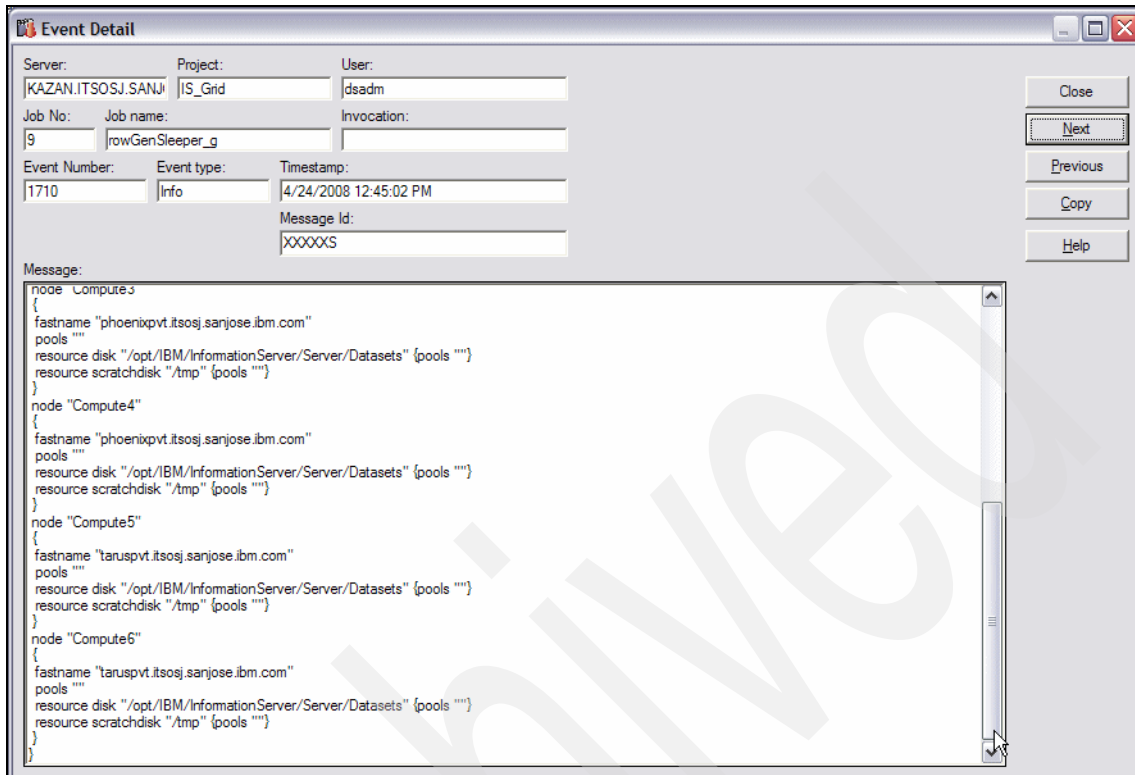


Figure 2-93 rowGenSleeper_g after failover, part 8 of 8

Example 2-148 Workload Scheduler LoadLeveler status output while the job was running on standby node nile after failure

Thu Apr 24 12:45:34 PDT 2008

Name	Schedd	InQ	Act	Startd	Run	LdAvg	Idle	Arch
OpSys								
luxor.itsosj.sanjose.ibm.	Avail	0	0	Busy	1	1.00	195	i386
Linux2								
nile.itsosj.sanjose.ibm.c	Avail	1	0	None	0	1.00	0	i386
Linux2								
phoenix.itsosj.sanjose.ib	Avail	0	0	Busy	1	0.00	9999	i386
Linux2								
tarus.itsosj.sanjose.ibm.	Avail	0	0	Busy	1	0.00	9999	i386
Linux2								
i386/Linux2	4 machines			1 jobs			3	running
Total Machines	4 machines			1 jobs			3	running

The Central Manager is defined on nile.itsosj.sanjose.ibm.com

The BACKFILL scheduler is in use

All machines on the machine_list are present.

Id	Owner	Submitted	ST	PRI	Class
Running On					

nile.109.0	dsadm	4/24 12:44	R	50	No_Class
phoenix					

1 job step(s) in queue, 0 waiting, 0 pending, 1 running, 0 held, 0 preempted

2.9 Step 5: Failing back to the original front end node ORION

Failback is the process of transferring the front end node role from the standby node to the original front end node orion.

The procedure is almost identical to the failover to the standby node procedure described in Section 2.8, “Step 4: Failing over to the standby node NILE” on page 221, except that the standby node (nile) is reverted back to a standby node role in controlled fashion.

Note: As mentioned earlier in 2.8, “Step 4: Failing over to the standby node NILE” on page 221, a number of scripts are provided with the installation of IBM Information Server which can be found in the `/etc/rc3.d` or `/etc/init.d` directories. They can be used without modification.

We developed the following scripts to simplify the takeover process from the front end node orion to the standby node nile:

1. The `failback_Nile.sh` script (shown in Example B-26 on page 294) performs an orderly shutdown of the standby node (nile) as follows:
 - a. Shuts down ganglia services
 - b. Shuts down LoadLeveler
 - c. Shuts down IBM Information Server
 - d. Shuts down NFS services
 - e. Unmounts the file systems
 - f. Shuts down the virtual IP addresses

Example 2-149 shows the output of the execution of this script.

Example 2-149 failback_Nile.sh script

```
[root@nile ~]# ./failback_Nile.sh
Checking if virtual IP addresses are still up...(this takes
approximately 3 seconds.)
    inet addr:9.43.86.77  Bcast:9.43.87.255  Mask:255.255.252.0
Setting current host as standby headnode...
Shutting down GANGLIA gmond:                [ OK ]
Stopping IBM LoadLeveler:                    [ OK ]
Stopping dstage services:                    [ OK ]
Stopping ISFserver services:                  [ OK ]
Stopping ISFagents services:                  [ OK ]
```

```

Stopping xmeta services: [ OK ]
gmetad (pid 29532) is running...
Stopping Ganglia gmetad, gmond, and httpd services...
Shutting down GANGLIA gmetad: [ OK ]
Stopping httpd: [ OK ]
Stopping NFS server
Shutting down NFS mountd: [ OK ]
Shutting down NFS daemon: [ OK ]
Shutting down NFS quotas: [ OK ]
Shutting down NFS services: [ OK ]
Unmounting SAN filesystems
Filesystem          1K-blocks      Used Available Use% Mounted on
/dev/mapper/VolGroup00-LogVol00
                    574278336    97650712 447455912   18% /
/dev/sda1            101086       15028      80839   16% /boot
none                 3628496         0    3628496   0% /dev/shm
Downing Virtual IP Addresses
eth0:1  Link encap:Ethernet HWaddr 00:02:55:AC:86:A4
        UP BROADCAST RUNNING MULTICAST MTU:1500 Metric:1

eth1:1  Link encap:Ethernet HWaddr 00:02:55:E4:5B:17
        UP BROADCAST RUNNING MULTICAST MTU:1500 Metric:1

```

2. The failback_Orion.sh script (shown in Example B-28 on page 297) executes on the original front end node (orion) and performs tasks similar to that of the failover_Nile.sh script.

Example 2-150 shows the output of the execution of this script.

Example 2-150 failback_Orion.sh script output

```

[root@orion ~]# ./failback_Orion.sh
Checking if virtual IP addresses are still up...(this takes
approximately 3 seconds.)
Setting current host as primary headnode...
Upping Virtual IP Addresses
eth0:1  Link encap:Ethernet HWaddr 00:14:5E:B4:16:3A
        inet addr:9.43.86.77 Bcast:9.43.87.255 Mask:255.255.252.0
        UP BROADCAST RUNNING MULTICAST MTU:1500 Metric:1
        Interrupt:169

eth1:1  Link encap:Ethernet HWaddr 00:14:5E:B4:16:3B
        inet addr:192.168.1.77 Bcast:192.168.3.255 Mask:255.255.252.0
        UP BROADCAST RUNNING MULTICAST MTU:1500 Metric:1
        Interrupt:169

```

```

Mounting SAN filesystems
Filesystem            1K-blocks      Used Available Use% Mounted on
/dev/mapper/VolGroup00-LogVol100
574536368 49624544 495727008 10% /
/dev/sda1            101086        15340      80527 17% /boot
none                4154816         0    4154816  0% /dev/shm
/dev/sdf1            196870980 32223380 154647140 18% /home
/dev/sdf2            98435476 13108964 80326284 15% /opt/IBM
/dev/sdf3            98435476    95748 93339500  1% /ds_overview
/dev/sdf5            71157244    554568 66988056  1% /opt/mqm
Setting LoadL_admin configuration file central_manager to orion
Restarting NFS server
Shutting down NFS mountd: [FAILED]
Shutting down NFS daemon: [FAILED]
Shutting down NFS quotas: [FAILED]
Shutting down NFS services: [ OK ]
Starting NFS services: [ OK ]
Starting NFS quotas: [ OK ]

```

3. The remount_Orion.sh script (shown in Example B-30 on page 301) also executes on the original front end node (orion) and performs the task of mounting the file systems on the compute nodes. This script is almost identical to the remount_Nile.sh script described earlier.

Example 2-147 on page 227 shows the output of the execution of this script.

Example 2-151 remount_Orion.sh script output

```

[root@orion ~]# ./remount_Orion.sh
phoenix
tarus
luxor
-----
kill resource tracker on the above compute nodes
<node name="phoenix" command="kill -9 $(lsdf 2> /dev/null | grep
'/opt/IBM' | grep 'resour' | awk '{print $2}' | sort | uniq) ">
</node>
<node name="tarus" command="kill -9 $(lsdf 2> /dev/null | grep
'/opt/IBM' | grep 'resour' | awk '{print $2}' | sort | uniq) ">
</node>
<node name="luxor" command="kill -9 $(lsdf 2> /dev/null | grep
'/opt/IBM' | grep 'resour' | awk '{print $2}' | sort | uniq) ">
</node>
-----
run umount -a

```

```

<node name="phoenix" command="umount -a ">
umount: /var/lib/nfs/rpc_pipefs: device is busy
umount: /: device is busy
</node>
<node name="tarus" command="umount -a ">
umount: /var/lib/nfs/rpc_pipefs: device is busy
umount: /: device is busy
</node>
<node name="luxor" command="umount -a ">
umount: /var/lib/nfs/rpc_pipefs: device is busy
umount: /: device is busy
</node>
-----
run df
<node name="phoenix" command="df ">
Filesystem            1K-blocks      Used Available Use% Mounted on
/dev/mapper/VolGroup00-LogVol100
                        574536368    2734972 542616580    1% /
</node>
<node name="tarus" command="df ">
Filesystem            1K-blocks      Used Available Use% Mounted on
/dev/mapper/VolGroup00-LogVol100
                        574536368    2746056 542605496    1% /
</node>
<node name="luxor" command="df ">
Filesystem            1K-blocks      Used Available Use% Mounted on
/dev/mapper/VolGroup00-LogVol100
                        138851328    80519320 51278696    62% /
</node>
-----
sleep for 3 minutes
-----
run mount -a
<node name="phoenix" command="mount -a ">
</node>
<node name="tarus" command="mount -a ">
</node>
<node name="luxor" command="mount -a ">
</node>
-----
run df
<node name="phoenix" command="df ">
Filesystem            1K-blocks      Used Available Use% Mounted on
/dev/mapper/VolGroup00-LogVol100
                        574536368    2734976 542616576    1% /

```

```

/dev/sda1          101086      13644      82223  15% /boot
none              4154816         0   4154816   0% /dev/shm
192.168.1.77:/home 196871008 32223840 154646688 18% /home
192.168.1.77:/ds_overview
                  98435488      95744   93339520   1% /ds_overview
192.168.1.77:/opt/IBM
                  98435488  13109120   80326144  15% /opt/IBM
</node>
<node name="tarus" command="df ">
Filesystem      1K-blocks      Used Available Use% Mounted on
/dev/mapper/VolGroup00-LogVol00
                  574536368   2746064 542605488   1% /
/dev/sda1        101086      13643      82224  15% /boot
none            4154816         0   4154816   0% /dev/shm
192.168.1.77:/home 196871008 32223840 154646688 18% /home
192.168.1.77:/ds_overview
                  98435488      95744   93339520   1% /ds_overview
192.168.1.77:/opt/IBM
                  98435488  13109120   80326144  15% /opt/IBM
</node>
<node name="luxor" command="df ">
Filesystem      1K-blocks      Used Available Use% Mounted on
/dev/mapper/VolGroup00-LogVol00
                  138851328  80519324 51278692   62% /
/dev/sda1        101086      12393      83474  13% /boot
none            3628544         0   3628544   0% /dev/shm
192.168.1.77:/home 196871008 32223840 154646688 18% /home
192.168.1.77:/ds_overview
                  98435488      95744   93339520   1% /ds_overview
192.168.1.77:/opt/IBM
                  98435488  13109120   80326144  15% /opt/IBM
</node>

```

We reran the rowGenSleepers_g job that we ran after failover (Figure 2-86 on page 230) to show that the original front end node (orion) had successfully taken over the front-end role again.

Example 2-152 on page 241 shows the output of the **llmonitor** command while this job is running. It shows one job running on three nodes (Startd column shows Busy), and the Central Manager being defined on orion.itsosj.sanjose.ibm.com.

Thu Apr 24 16:48:32 PDT 2008

Name	Schedd	InQ	Act	Startd	Run	LdAvg	Idle	Arch
OpSys								
luxor.itsosj.sanjose.ibm.	Avail	0	0	Busy	1	1.02	9999	i386
Linux2								
orion.itsosj.sanjose.ibm.	Avail	0	0	None	0	0.15	142	i386
Linux2								
phoenix.itsosj.sanjose.ib	Avail	0	0	Busy	1	0.00	9999	i386
Linux2								
tarus.itsosj.sanjose.ibm.	Avail	0	0	Busy	1	0.02	9999	i386
Linux2								
i386/Linux2	4 machines		0	jobs		3	running	
Total Machines	4 machines		0	jobs		3	running	

The Central Manager is defined on orion.itsosj.sanjose.ibm.com

The BACKFILL scheduler is in use

All machines on the machine_list are present.

Id	Owner	Submitted	ST	PRI	Class
Running On					

orion.110.0	dsadm	4/24 16:47	R	50	No_Class
phoenix					

1 job step(s) in queue, 0 waiting, 0 pending, **1 running**, 0 held, 0 preempted

Grid deployment toolkits

In this appendix we provide an overview of the Build your Own Grid (BYOG) toolkit for setting up and configuring a Red Hat or SUSE Linux Grid environment infrastructure, and the Grid Enablement Toolkit for configuring a Red Hat or SUSE Linux Grid environment.

A.1 The toolkits

The following two toolkits will become available for free download for deploying a grid solution with IBM Information Server:

- ▶ BYOG toolkit
- ▶ Grid Enablement Toolkit

Note: These toolkits are continually being enhanced in response to new requirements based on customer feedback as well as IBM Information Server product enhancements. We are currently working on providing a link to up-to-date sources of information about these toolkits. Until such a download site becomes available, you must contact your local sales representative to obtain these toolkits. In this appendix, we focus on providing an overview of the core functionality of these toolkits.

The two toolkits support versions 7.5.1 onward of DataStage, QualityStage, and ProfileStage.

A.2 Build Your Own Grid toolkit

The BYOG toolkit is a set of scripts and templates designed to help configure a RedHat or SUSE Linux Grid environment. It provides the tools necessary to build the front end node as well as the compute nodes without system administration on each compute node.

The BYOG toolkit is available as a tar file (BYOG.tar), and it contains the following components:

- The scripts and files as shown in Table A-1.

Table A-1 Scripts and files provided in the BYOG.tar file

Script name	Description
PXE_cfg.sh	Creates PXE boot configuration files based on the range of IP addresses configured in the /etc/dhcpd.conf file. It also creates pointer .cfg file to the actual kickstart file.
create_ks.sh	Redhat Linux only Creates kickstart files based on a kickstart_template file and dhcpd.conf configuration files. The template file contains the base name for the compute nodes as follows: -hostname compute-# Replace # with the host name based on the PXE boot sequence
create_Al.sh	SUSE Linux only Corresponds to the create_ks.sh script and creates the AutoInstall.xml files based on a AutoInstall_template file and the dhcpd.conf configuration files.
Define_ip.sh	Reads /var/log/messages files to re-create /etc/dhcpd.conf and /etc/hosts file based on the MAC address and the DHCP assigned IP address.
post-install/Master	Part of the %post processing of kickstart processing. The Master script runs additional processing steps for each compute node after the initial build. The current processed order is NIS and NFS.
post-install/NFS	Adds any static NFS mounts to the /etc/fstab file
post-install/NIS	Configures NIS on the compute nodes based on information provided in the postenv file
post-install/post-env file	Environment file that identifies Frontend hostname, NIS domain and other environment variables used within all post-install scripts.

- A document providing information about sizing, options, installation, and configuration of the physical grid environment infrastructure for the RedHat Linux or SUSE Linux platforms.

It includes installation and configuration of resource managers such as LoadLeveler, SGE, Platform LSF and PBSpro & Torque, and resource monitors such as Ganglia, Big Brother, Parmom, Performance co-Pilot, and Clumon.

The table of contents include the following:

- Introductions
 - Grid definitions
 - Clustered versus Grid configurations
- Typical Grid environments
 - NAS (Network Attached Storage)
 - SAN (Storage Area Networks) or directly connected devices
 - Platform specifications
 - Architectural configurations
- Installation of BYOG tools
 - Linux package installation
 - Toolkit contents
 - Installation instructions
- System Management
 - User Management
 - Linux PXE Booting
 - Configuration of DHCP
 - tftp Service configuration
 - NFS configuration
 - Solaris™ Jumpstart
- Resource Management
 - IBM Loadleveler
 - SGE
 - Platform LSF
 - PBSpro & Torque
- Resource Monitor
 - Ganglia
 - Big Brother
 - Parmom
 - Performance Co-pilot
 - Clumon (Clumon.nasca.uiuc.edu)
- SSH and RSH configuration

- Troubleshooting and Monitoring
 - Jobs do not get released from Resource Manager
 - Resource Manager indicates running, no apt_config_file
 - How to see what's queued and why
 - SGE does not start on front end

A.3 Grid Enablement Toolkit

The Grid Enablement Toolkit is a set of scripts and templates whose goal is to create a dynamic configuration file (APT_CONFIG_FILE) using any one of several supported resource managers which identifies nodes that are available for processing.

The Grid Enablement Toolkit is available as a tar file (grid_enabled<version>.tar), and it contains the following components:

- Scripts and files as shown in Table A-2.

After running the install.sh script, some important files are created/updated which are also listed here.

Table A-2 Scripts and files provided in the grid_enabled.4.0.1.tar file

Script name	Description
install.sh	Installs the Grid Enablement Toolkit by prompting for default values and placing the osh intercept script into the \$APT_ORCHHOME/bin directory. For ProfileStage, the applications ColumnAnalysis, TableAnalysis and CrossTableAnalysis are replaced with an interceptor script that calls the Grid Enablement components prior to running the original application.
README.txt	Contains information about basic design and changes by release.
osh.sh	Renamed to osh after changing the original osh to osh.exe in the \$APT_ORCHHOME/bin directory. The script will either make direct calls to the osh.exe or the osh_conductor script in the grid_enabled directory.
osh_conductor	Called once for each job to determine if a dynamic configuration is required. Performs some initial validation of rules and then runs the original osh.exe

Script name	Description
Dynamic_grid.sh	Submits a job to a Resource Manager that returns nodes available for processing. The environment variable APT_CONFIG_FILE is updated to reflect the new configuration file generated based on the nodes returned from the Resource Manager. The syntax for the Resource Manager submission is based on installation and defined as "Engine=" within the grid_global_values file.
grid_global_values file	Configuration values based on information prompted for during the installation.
grid_config.apt file	Template configuration file used to construct the APT_CONFIG_FILE with Resource Manager information.
qs_modPar.sh	For 7.x QualityStage only Used to modify a deployed QualityStage .par script. Running this script will Grid enable the .par script and allow additional command line functions not available within the QualityStage GUI.
PS_AnalysisIntercept.sh	For 7.X ProfileStage only Intercept script to Grid enable Applications within ProfileStage
orchadmin.sh	Replaces the default orchadmin executable within \$APT_ORCHHOME/bin. Used to make sure all processing uses the newly constructed config file instead of the configuration file stored within a dataset.

- A document that provides a set of standard practices for design, development, and implementation of Grid solutions for the Parallel components of IBM Information Server such as DataStage, QualityStage, and Information Analyzer for the RedHat Linux or SUSE Linux platforms.

The table of contents include the following:

- Introduction
 - Grid definitions
 - Clustered versus Grid configuration
- Parallel Framework architecture overview
 - Runtime architecture
- Typical Grid environments
 - NAS (Network Attached Storage)
 - SAN (Storage Area Networks) or directly connected devices
 - Platform specifications
 - Architectural configurations

- Resource Management
 - IBM Loadleveler
 - SGE
 - Platform LSF
 - PBSpro/Torque/OpenPBS
 - Datasynapse Gridserver
- Installing application patches
 - Parallel Framework of Information Server
 - Qualitystage Ee (7.x Only)
 - Profilestage Ee (7.x Only)
- Installation of Grid components
 - Prerequisites
 - Toolkit Contents
 - Installation worksheet
 - Installation instructions
 - Debugging why test.sh is not running as expected
 - Template grid_config.apl file
 - Global configuration file grid_global_values
 - Node translation
 - Compute nodes require NFS mounts
- Toolkit Environment Variables
 - Information Server admin parameters
 - APT_GRID_ENABLE
 - APT_GRID_QUEUE
 - APT_GRID_COMPUTENODES
 - APT_GRID_PARTITIONS
 - APT_GRID_FROM_NODES (old releases only)
 - APT_GRID_FROM_PARTITIONS (old releases only)
 - APT_GRID_OPTS
 - APT_GRID_SEQFILE_HOST
 - APT_GRID_SEQFILE_HOST2
 - APT_GRID_CONFIG
 - APT_GRID_IDENTIFIER
 - APT_GRID_STAT_CMD
 - APT_GRID_SCRIPTPRE
 - APT_GRID_SCRIPTPOST

- Using the Grid Enablement Toolkit
 - Within a Datastage or Qualitystage Ee job
 - Using Sequential File Host Values
 - Datastage Job Sequence
 - Parallel Job Sequence
 - Time files
- Datastage Node Pool constraints
 - Default apt_config_file modifications
 - Modification of “Node Pool” constraint
 - Actual apt_config_file generated
- Qualitystage 7.x requirements
 - Installation
 - Access to Enterprise Edition engine
 - Qualitystage 7.x grid enablement
 - How to grid enable a Qualitystage .par file
- Profilestage 7.x requirements
 - Installation
 - Access to Enterprise Edition engine
- Troubleshooting and Monitoring
 - Jobs do not get released from resource manager
 - Resource Manager indicates running, no apt_config_file
 - How to see what's queued and why
 - SGE does not start on front end
 - Java core dumps / no configuration file created
- Appendix A: SSH and RSH configuration



Commands and scripts used in the migration scenario

In this appendix we document some of the commands and scripts used in the migration scenario.

B.1 Introduction

This appendix documents some of the commands and scripts used in the migration scenario as follows:

- ▶ Post install scripts
- ▶ Workload Scheduler Load Leveler commands
- ▶ Grid toolkit resource manager scripts
- ▶ Scripts developed during the residency

These scripts are briefly described in the following sections.

B.2 Post install scripts

Example B-1 shows the modified NFS script (where the default mount points are changed to the custom mount points) invoked by the Master script in the PXEBoot execution as part of the post install process. The NFS script is provided as part of the BYOG toolkit and mounts file systems shared between the front end node and the standby node packages.

Example: B-1 NFS script

```
date

if [ "$FRONTEND_IP" = "" ] ; then
    echo Invalid front-end IP address. Set FRONTEND_IP environment
    variable within post-env
    exit
fi

cat << EOF >> /etc/fstab
${FRONTEND_IP}:/home /home nfs rw 0 0
${FRONTEND_IP}:/ds_overview /ds_overview nfs rw 0 0
${FRONTEND_IP}:/opt/IBM /opt/IBM nfs rw 0 0
EOF

df -k

mkdir -v -p /ds_overview /home /opt/IBM
mount ${FRONTEND_IP}:/opt/IBM
mount ${FRONTEND_IP}:/ds_overview
mount ${FRONTEND_IP}:/home

df -k

date
```

B.3 Workload Scheduler LoadLeveler commands

The Workload Scheduler LoadLeveler commands are as follows:

- `llstatus`
- `llq`

We used these commands as follows:

- The **`llstatus`** command returns status information about machines in the LoadLeveler cluster as shown in Example B-2.

Example: B-2 llstatus command

```
[dsadm@orion ~]$ llstatus
Name                               Schedd  InQ Act Startd Run LdAvg Idle Arch
OpSys
luxor.itsosj.sanjose.ibm. Avail      0  0 Busy   1 1.23 3456 i386
Linux2
orion.itsosj.sanjose.ibm. Avail      2  1 None    0 0.01  56 i386
Linux2
phoenix.itsosj.sanjose.ib Avail      0  0 Busy   1 0.01 2269 i386
Linux2
tarus.itsosj.sanjose.ibm. Avail      0  0 Busy   1 0.00 2665 i386
Linux2

i386/Linux2                       4 machines      2 jobs      3 running
Total Machines                    4 machines      2 jobs      3 running
```

The Central Manager is defined on orion.itsosj.sanjose.ibm.com

The BACKFILL scheduler is in use

All machines on the machine_list are present.

- The **`llq`** command returns information about jobs in the LoadLeveler queues as shown in Example B-3. The job name is not shown in the output.

Example: B-3 llq command

```
[dsadm@orion ~]$ llq
Id                               Owner      Submitted   ST PRI Class
Running On
-----
orion.84.0                       dsadm      4/16 17:51 R  50  No_Class
tarus
```

```
orion.85.0                dsadm          4/16 17:55 I  50  No_Class

2 job step(s) in queue, 1 waiting, 0 pending, 1 running, 0 held, 0
preempted
```

For information about LoadLeveler commands, refer to *LoadLeveler for AIX 5L and Linux Version 3 Release 3.1: Using and Administering*, SA22-7881-04.

B.4 Grid toolkit resource manager commands

Scripts are provided with the grid toolkits to monitor resource manager resources. Because multiple resource managers are supported by the toolkits, these scripts not only provide resource manager independence by shielding the administrator from differences in the underlying commands, but also additional level of detail.

The grid toolkit resource manager commands are as follows:

- `listq`
- `canjob`

We used these commands as follows:

- The `listq` command returns information about jobs in the resource manager (LoadLeveler in our case) queues as shown in Example B-4.

To get more details on a waiting job, the `listq <Step Id>` command provides this as shown in Example B-5 on page 255.

Example: B-4 listq command

```
[dsadm@orion ~]$ listq
```

Step Id	Job Name	ST	PRI	Class
Running	On			

orion.85.0	rowGenSeqFile_28910	I	50	No_Class
orion.84.0	rowGenSeq2Files_g_27	R	50	No_Class
tarus				

```
2 job step(s) in queue, 1 waiting, 0 pending, 1 running, 0 held, 0
preempted
```

Example: B-5 Getting more details on waiting jobs

```
[dsadm@orion ~]$ listq orion.85.0
===== Job Step orion.itsosj.sanjose.ibm.com.85.0
=====
      Job Step Id: orion.itsosj.sanjose.ibm.com.85.0
      Job Name: rowGenSeqFile_28910
      Step Name: 0
      Structure Version: 10
      Owner: dsadm
      Queue Date: Wed 16 Apr 2008 05:55:50 PM PDT
      Status: Idle
      Reservation ID:
      Requested Res. ID:
      Scheduling Cluster:
      Submitting Cluster:
      Sending Cluster:
      Requested Cluster:
      Schedd History:
      Outbound Schedds:
      Submitting User:
      Execution Factor: 1
      Dispatch Time:
      Completion Date:
      Completion Code:
      Favored Job: No
      User Priority: 50
      user_sysprio: 0
      class_sysprio: 0
      group_sysprio: 0
      System Priority: -66640
      q_sysprio: -66640
      Previous q_sysprio: 0
      Notifications: Never
      Virtual Image Size: 3 kb
      Large Page: N
      Checkpointable: no
      Ckpt Start Time:
      Good Ckpt Time/Date:
      Ckpt Elapse Time: 0 seconds
      Fail Ckpt Time/Date:
      Ckpt Accum Time: 0 seconds
      Checkpoint File:
      Ckpt Execute Dir:
      Restart From Ckpt: no
```

```
Restart Same Nodes: no
  Restart: yes
  Preemptable: yes
Preempt Wait Count: 0
  Hold Job Until:
    RSet: RSET_NONE
Mcm Affinity Options:
  Env:
  In: /dev/null
  Out:
/home/dsadm/grid_jobdir/rowGenSeqFile_28910.stdout
  Err:
/home/dsadm/grid_jobdir/rowGenSeqFile_28910.stderr
Initial Working Dir: /home/dsadm/grid_jobdir
  Dependency:
  Resources:
  Step Type: General Parallel
  Node Usage: shared
Submitting Host: orion.itsosj.sanjose.ibm.com
Schedd Host: orion.itsosj.sanjose.ibm.com
Job Queue Key:
  Notify User: dsadm@orion.itsosj.sanjose.ibm.com
  Shell: /bin/bash
LoadLeveler Group: No_Group
  Class: No_Class
  Ckpt Hard Limit: undefined
  Ckpt Soft Limit: undefined
  Cpu Hard Limit: undefined
  Cpu Soft Limit: undefined
  Data Hard Limit: undefined
  Data Soft Limit: undefined
  Core Hard Limit: undefined
  Core Soft Limit: undefined
  File Hard Limit: undefined
  File Soft Limit: undefined
  Stack Hard Limit: undefined
  Stack Soft Limit: undefined
  Rss Hard Limit: undefined
  Rss Soft Limit: undefined
Step Cpu Hard Limit: undefined
Step Cpu Soft Limit: undefined
Wall Clk Hard Limit: 00:30:00 (1800 seconds)
Wall Clk Soft Limit: 00:30:00 (1800 seconds)
  Comment:
  Account:
```

```
        Unix Group: dstage
        NQS Submit Queue:
        NQS Query Queues:
Negotiator Messages:
        Bulk Transfer: No
Step Adapter Memory: 0 bytes
Adapter Requirement:
        Step Cpus: 0
Step Virtual Memory: 0.000 mb
        Step Real Memory: 0.000 mb
```

```
-----
Node
----
```

```

Name          :
Requirements   : (Arch == "i386") && (OpSys == "Linux2")
Preferences    :
Node minimum   : 1
Node maximum   : 1
Node actual    : 0
Allocated Hosts :
```

```
Master Task
-----
```

```

Executable    : /home/dsadm/grid_jobdir/rowGenSeqFile_28910.sh
Exec Args     :
Num Task Inst: 0
Task Instance:
```

```
Task
----
```

```

Num Task Inst: 0
Task Instance:
```

```
===== EVALUATIONS FOR JOB STEP
orion.itsosj.sanjose.ibm.com.85.0 =====
```

```
The class of this job step is "No_Class".
Total number of available initiators of this class on all machines
in the cluster: 0
Minimum number of initiators of this class required by job step: 1
```

The number of available initiators of this class is not sufficient for this job step.
Not enough resources to start now:
This step is top-dog.
Considered at: Wed Apr 16 18:08:13 2008
Will start by: Wed Apr 16 18:21:11 2008

- The **canjob <Step Id>** command cancels a job in the resource manager (LoadLeveler in our case) queue as shown in Example B-6.
The Director log output of this cancelled job is shown in Example B-7 on page 259.

Example: B-6 canjob command (cancel a waiting job)

```
[dsadm@orion ~]$ listq
```

Step Id	Job Name	ST	PRI	Class
Running On				
-----	-----	---	---	-----

orion.85.0	rowGenSeqFile_28910	I	50	No_Class
orion.84.0	rowGenSeq2Files_g_27	R	50	No_Class
tarus				

2 job step(s) in queue, 1 waiting, 0 pending, 1 running, 0 held, 0 preempted

```
[dsadm@orion grid_jobdir]$ canjob orion.85.0
```

```
orion.85.0 rowGenSeqFile_28910
```

```
cancel job (orion.85.0)? yes
```

```
llcancel: Cancel command has been sent to the central manager.
```

```
[dsadm@orion grid_jobdir]$ listq
```

Step Id	Job Name	ST	PRI	Class
Running On				
-----	-----	---	---	-----

orion.84.0	rowGenSeq2Files_g_27	R	50	No_Class
tarus				

1 job step(s) in queue, 0 waiting, 0 pending, 1 running, 0 held, 0 preempted

Example: B-7 DataStage Director log relating to the cancelled job orion.85.0

Server:KAZAN.ITSOSJ.SANJOSE.IBM.COM
Project:IS_Grid
Job No:7
Job name:rowGenSeqFile
Invocation:
Event Number:148
Event type:Info
User:dsadm
Timestamp:16/04/2008 6:10:54 PM
Message Id:XXA1805
Message:
<Dynamic_grid.sh> Waiting for Release from Queue

Server:KAZAN.ITSOSJ.SANJOSE.IBM.COM
Project:IS_Grid
Job No:7
Job name:rowGenSeqFile
Invocation:
Event Number:149
Event type:Info
User:dsadm
Timestamp:16/04/2008 6:12:39 PM
Message Id:XXA1810
Message:
<Dynamic_grid.sh> Waiting for Release from Queue
<Dynamic_grid.sh> Job cancelled.

Server:KAZAN.ITSOSJ.SANJOSE.IBM.COM
Project:IS_Grid
Job No:7
Job name:rowGenSeqFile
Invocation:
Event Number:150
Event type:Fatal
User:dsadm
Timestamp:16/04/2008 6:12:39 PM
Message Id:DSTAGE_RUN_E_0372
Message:
Parallel job reports failure (code 1)

B.5 Scripts developed

We developed a number of scripts (miscellaneous, services, and HA) to ease the installation, administration, and management of our grid environment. These are briefly described here.

B.5.1 Miscellaneous scripts

We developed the following miscellaneous scripts:

- ▶ llmonitor
- ▶ nisupdatefiles.sh
- ▶ llSServer.sh
- ▶ runcmd.sh

We used these scripts as follows:

- ▶ llmonitor script issues the date, llstatus and listq commands at three second intervals as shown in Example B-8.
 - Example B-9 on page 261 shows the output of this script when no jobs are running.
 - Example B-10 on page 262 shows the output of this script with jobs running.

Example: B-8 llmonitor.sh script

```
#!/bin/bash
# llmonitor - script to monitor LoadLeveler nodes and queue
#
# Written by Anthony Corrente
# Date: Wed Mar 19 14:28:24 PST 2008
#
```

```
DS=$(date +%Y%m%d%H%M%S)
while :
do
    echo
    "-----"
    date
    echo
    "-----"
    llstatus
```

```

echo
listq
echo
"-----"
-----"
echo
sleep 3
done | tee /tmp/llmonitor.${DS?}.p$.log

```

Example: B-9 llmonitor output with no jobs running

Tue Apr 15 16:34:08 PDT 2008

Name	Schedd	InQ	Act	Startd	Run	LdAvg	Idle	Arch	OpSys
luxor.itsosj.san jose.ibm.	Avail	0	0	None	0	1.05	0	i386	Linux2
phoenix.itsosj.san jose.ib	Avail	0	0	Idle	0	0.03	791	i386	Linux2
tarus.itsosj.san jose.ibm.	Avail	0	0	Idle	0	0.00	1862	i386	Linux2
i386/Linux2	3 machines			0 jobs	0	running			
Total Machines	3 machines			0 jobs	0	running			

The Central Manager is defined on luxor.itsosj.san jose.ibm.com

The BACKFILL scheduler is in use

All machines on the machine_list are present.

llq: There is currently no job status to report.

Tue Apr 15 16:34:11 PDT 2008

Name	Schedd	InQ	Act	Startd	Run	LdAvg	Idle	Arch	OpSys
luxor.itsosj.san jose.ibm.	Avail	0	0	None	0	1.05	0	i386	Linux2
phoenix.itsosj.san jose.ib	Avail	0	0	Idle	0	0.03	791	i386	Linux2
tarus.itsosj.san jose.ibm.	Avail	0	0	Idle	0	0.00	1862	i386	Linux2
i386/Linux2	3 machines			0 jobs	0	running			
Total Machines	3 machines			0 jobs	0	running			

The Central Manager is defined on luxor.itsosj.san jose.ibm.com

The BACKFILL scheduler is in use

All machines on the machine_list are present.

llq: There is currently no job status to report.

Example: B-10 llmonitor output with jobs running

Tue Apr 15 16:35:15 PDT 2008

Name	Schedd	InQ	Act	Startd	Run	LdAvg	Idle	Arch	OpSys
luxor.itsosj.sanjose.ibm.	Avail	0	0	None	0	1.05	0	i386	Linux2
phoenix.itsosj.sanjose.ib	Avail	0	0	Idle	0	0.00	911	i386	Linux2
tarus.itsosj.sanjose.ibm.	Avail	0	0	Idle	0	0.00	1982	i386	Linux2
i386/Linux2	3 machines		0	jobs	0	running			
Total Machines	3 machines		0	jobs	0	running			

The Central Manager is defined on luxor.itsosj.sanjose.ibm.com

The BACKFILL scheduler is in use

All machines on the machine_list are present.

llq: There is currently no job status to report.

Tue Apr 15 16:35:18 PDT 2008

Name	Schedd	InQ	Act	Startd	Run	LdAvg	Idle	Arch	OpSys
luxor.itsosj.sanjose.ibm.	Avail	0	0	None	0	1.05	0	i386	Linux2
phoenix.itsosj.sanjose.ib	Avail	0	0	Idle	0	0.00	911	i386	Linux2
tarus.itsosj.sanjose.ibm.	Avail	0	0	Busy	1	0.00	2042	i386	Linux2
i386/Linux2	3 machines		0	jobs	1	running			
Total Machines	3 machines		0	jobs	1	running			

The Central Manager is defined on luxor.itsosj.sanjose.ibm.com

The BACKFILL scheduler is in use

All machines on the machine_list are present.

Step Id	Job Name	ST	PRI	Class	Running On
luxor.239.0	testjob_18245	R	50	medium	tarus

- nisupdatefiles.sh script (shown in Example B-11 on page 263) does a backup of current local files related to user and hosts (such as /etc/hosts, /etc/shadow, /etc/passwd, /etc/group) and updates them with information from the NIS server database. This script is meant to be executed on the NIS subordinate server to update the local files (/etc/hosts, /etc/shadow, /etc/passwd, /etc/group) to sync with the NIS server database information.

Note: The NIS master server database and the NIS subordinate server database is kept in sync through the **make -C /var/yp** command (described in Example 2-47 on page 85) executed on the NIS master server.

Example: B-11 nisupdatefiles.sh script

```
#!/bin/bash
# nisupdatefile.sh
### set enviables ###
DS=$(date +%Y%m%d%H%M)
### Make a baseline backup of the target files ###

[ ! -f /etc/hosts.baseline ] && cp /etc/hosts /etc/hosts.baseline
[ ! -f /etc/passwd.baseline ] && cp /etc/passwd /etc/passwd.baseline
[ ! -f /etc/shadow.baseline ] && cp /etc/shadow /etc/shadow.baseline
[ ! -f /etc/group.baseline ] && cp /etc/group /etc/group.baseline

### dump NIS db and create the files into a work directory ###

/usr/lib/yp/makedbm -u /var/yp/IS.redbook.grid/passwd.byname | awk
-F':' '{split($1,userid," ");print userid[1] "x:" $3 ":" $4 ":" $5 ":"
$6 ":" $7}' | grep -v '^YP'|sort > /tmp/passwd

/usr/lib/yp/makedbm -u /var/yp/IS.redbook.grid/passwd.byname | awk
-F':' '{split($1,userid," ");print userid[1] ":" $2
":13539:0:99999:7:::}" | grep -v '^YP'|sort > /tmp/shadow

/usr/lib/yp/makedbm -u /var/yp/IS.redbook.grid/group.byname | grep -v
'^YP'|awk '{print $2}'|sort > /tmp/group

/usr/lib/yp/makedbm -u /var/yp/IS.redbook.grid/hosts.byname| grep -v
'^YP'| grep -v 'localhost'|awk '{print $2" "$3" "$4}'|sort|uniq >
/tmp/hosts

### make a copy of the existing target file using a datestamp ###

cp /etc/hosts /etc/hosts.$DS.bak
cp /etc/shadow /etc/shadow.$DS.bak
cp /etc/passwd /etc/passwd.$DS.bak
cp /etc/group /etc/group.$DS.bak

### Combine the new files with the baseline files ###
```

```
cat /etc/hosts.baseline /tmp/hosts > /etc/hosts
cat /etc/passwd.baseline /tmp/passwd > /etc/passwd
cat /etc/shadow.baseline /tmp/shadow > /etc/shadow
cat /etc/group.baseline /tmp/group > /etc/group
```

Place this script in some location (/root in our case) and run in a cron job. To install the above script in the crontab, use the following command:

```
# crontab -e
# Run the script to update the local files from NIS server.
# The following will run everyday at 2:15AM.
15 2 * * * /root/nisupdatefiles.sh
```

Note: You must choose an update cycle based on the expected rate of changes to user accounts.

- IIServer.sh script (shown in Example B-12) starts up all the services required for the IBM Information Server grid environment.

Example: B-12 IIServer.sh script

```
#!/bin/bash
#
# IIServer      Startup script for IBM Information Server Grid
# Written by    Anthony Corrente
#
# chkconfig: 5 99 1
# description: This script is used to start/stop/query the status \
#               of the IBM Information Server (R) on Grid.

# Source function library.
. /etc/rc.d/init.d/functions

echo_running() {
    [ "$BOOTUP" = "color" ] && $MOVE_TO_COL
    echo -n "[ "
    [ "$BOOTUP" = "color" ] && $SETCOLOR_SUCCESS
    echo -n "$RUNNING"
    [ "$BOOTUP" = "color" ] && $SETCOLOR_NORMAL
    echo -n " ]"
    echo -ne "\r"
    return 0
}
```

```

echo_stopped() {
    [ "$BOOTUP" = "color" ] && $MOVE_TO_COL
    echo -n "[ "
    [ "$BOOTUP" = "color" ] && $SETCOLOR_FAILURE
    echo -n "$STOPPED"
    [ "$BOOTUP" = "color" ] && $SETCOLOR_NORMAL
    echo -n " ]"
    echo -ne "\r"
    return 0
}

REVAL=0

start() {
    RETVAL=0
    for service in xmeta ISFagents ISFserver dstage LoadL gmond
    do
        service ${service?} start
        let RETVAL=RETVAL+$?
    done
    [ $RETVAL -eq 0 ] && touch /var/lock/subsys/IISGrid
    return $RETVAL
}

stop () {
    RETVAL=0
    for service in gmond LoadL dstage ISFserver ISFagents xmeta
    do
        service ${service?} stop
        let RETVAL=RETVAL+$?
    done
    [ $RETVAL -eq 0 ] && rm -rf /var/lock/subsys/IISGrid
    return $RETVAL
}

echo_status () {
    if [ $RETVAL -eq 0 ]; then
        echo_running
    else
        echo_stopped
    fi
}

status () {

```

```

iisRETVAL=0
for service in xmeta ISFagents ISFserver dstage LoadL gmond
do
    RETVAL=0
    servicelog=$(service ${service?} status)
    RETVAL=$?
    echo -n "Status for ${service?} is"
    let iisRETVAL=iisRETVAL+RETVAL
    echo_status
    echo
    [ $VERBOSE -eq 1 ] && (echo ${servicelog}; echo)
done
echo -n "Getting Information Server Grid status: "
RETVAL=$iisRETVAL
echo_status
echo
return $iisRETVAL
}

VERBOSE=0
if [ "verbose" == "$1" ]; then
    VERBOSE=1
    shift
fi

# See how we were called.

case "$1" in
    start)
        start
        ;;
    stop)
        stop
        ;;
    status)
        status
        ;;
    restart)
        stop
        start
        ;;
    condrestart)
        if [ -f /var/lock/subsys/IISGrid ]; then

```



```

        stop
        start
    fi
    ;;
*)
    echo $"Usage: $0 [verbose] {start|stop|status|restart|condrestart}"
    exit 1
esac

exit $RETVAL

```

- `runcmd.sh` script (shown in Example B-13) executes specified commands and writes the output to a file. This script is invoked by the `remount_Nile.sh` and `remount.Orion.sh` scripts.

Example: B-13 `runcmd.sh` script

```

#!/bin/sh
CMD=$1
shift
OPTS=$*
for h in $(cat nodes.cfg)
do
    printf "<node name=\"$h\" command=\"${CMD} ${OPTS}\">\n"
    ssh ${h} "${CMD} ${OPTS}"
    printf "</node>\n"
done | tee /tmp/root_runcmd.$$out

```

B.5.2 Services scripts

The `IIServer.sh` script (Example B-12 on page 264) uses the following services:

- `dstage.sh` script shown in Example B-14 on page 268.
- `ISFagents.sh` script shown in Example B-15 on page 270.
- `ISFserver.sh` script shown in Example B-16 on page 274.
- `IISGrid.sh` script shown in Example B-17 on page 277.
- `xmeta.sh` script shown in Example B-18 on page 280.
- `Loadl.sh` script shown in Example B-19 on page 283.

Two additional scripts, `install.sh` (shown in Example B-20 on page 286) and `uninstall.sh` (shown in Example B-22 on page 288,) are provided.

All these scripts (including `IIServer.sh`) must be placed in the `/opt/IBM/HA_Common/init.d` directory.

These services (including IIServer.sh) may be installed by executing the install.sh script.

Note: These scripts may be uninstalled by executing the uninstall.sh script

The usage of the IIServer service is as follows:

```
/etc/init.d/IIServer [verbose] {start|stop|status|restart|reload|condrestart}
```

Sample output of various options of the execution of the IIServer service is shown in Example B-21 on page 287.

Example: B-14 dstage.sh script

```
#!/bin/bash
#
# dstage          Startup script for dstage
# Written by      Anthony Corrente
#
# chkconfig: 5 99 1
# description: dstage is a service that is used by Information Server \
#              to start/stop/query the status of the DataStage engine.
# pidfile: /var/run/dstage.pid

# Source function library.
. /etc/rc.d/init.d/functions

#
initialize() {
    VERBOSE=0
    RETVAL=0
    dsuser="dsadm"
    dshome=$(cat /.dshome)
    dsrc=${dshome?}/sample/ds.rc
    dsstop="${dsrc?} stop"
    dsstart="${dsrc?} start"
    dsinfo="\${DSHOME}/uv/bin -admin -info"
    dsproc=dsrpcd
    dslog=/tmp/dstage.$$log
}

dspid() {
    pidof ${dsproc?}
}

retval() {
```

```

        if [ $RETVAL -eq 0 ]; then
            echo_success
        else
            echo_failure
        fi
    }

    echo_log() {
        [ $VERBOSE -eq 1 ] && (cat ${dslog?}; rm -rf ${dslog?})
    }

    start() {
        echo -n "Starting dstage services: "
        status ${dsproc?} 2>&1 > ${dslog?}
        RETVAL=$?
        if [ $RETVAL -ne 0 ]; then
            ${dsstart?} 2>&1 > ${dslog?}
            RETVAL=$?
            [ $RETVAL -eq 0 ] && touch /var/lock/subsys/dstage && dspid >
/var/run/dstage.pid
        fi
        retval
        echo
        echo_log
        return $RETVAL
    }

    stop() {
        echo -n "Stopping dstage services: "
        status ${dsproc?} 2>&1 > ${dslog?}
        RETVAL=$?
        if [ $RETVAL -eq 0 ]; then
            ${dsstop?} 2>&1 > ${dslog?}
            RETVAL=$?
            [ $RETVAL -eq 0 ] && rm -f /var/lock/subsys/dstage && rm -rf
/var/run/dstage.pid
        fi
        retval
        echo
        echo_log
        return $RETVAL
    }

    initialize
    if [ "verbose" == "$1" ]; then

```

```

        VERBOSE=1
        shift
    fi

    # See how we were called.
    case "$1" in
        start)
            start
            ;;
        stop)
            stop
            ;;
        status)
            echo -n "Getting dstage status: "
            status ${dsproc}
            RETVAL=$?
            ;;
        restart)
            stop
            start
            ;;
        condrestart)
            if [ -f /var/lock/subsys/dstage ]; then
                stop
                start
            fi
            ;;
        *)
            echo "Usage: $0 [verbose] {start|stop|status|restart|condrestart}"
            exit 1
    esac

    exit $RETVAL

```

Example: B-15 ISFagents.sh script

```

#!/bin/bash
#
# ISFagents      Startup script for ISFagents
# Written by     Anthony Corrente
#
# chkconfig: 5 97 3
# description: ISFagents is a service that is used by Information
Server \

```

```

#           to start/stop/query the status of the
\
#           Information Services Framework agents.
# pidfile: /var/run/ISFagents.pid

# Source function library.
. /etc/rc.d/init.d/functions

#

initialize() {
    VERBOSE=0
    RETVAL=0
    IS_INIT_D=true
    export IS_INIT_D
    agenthome=/opt/IBM/InformationServer/ASBNode
    nodeagents="${agenthome?}/bin/NodeAgents.sh"
    agentstart="${nodeagents?} start"
    agentstop="${nodeagents?} stop"
    agentproc=${agenthome?}/apps/jre/java
    agentlog=/tmp/ISFagents.$$log
}

retval() {
    if [ $RETVAL -eq 0 ]; then
        echo_success
    else
        echo_failure
    fi
}

echo_log() {
    [ $VERBOSE -eq 1 ] && (cat ${agentlog?}; rm -rf ${agentlog?})
}

agentpids(){
    while [ ! -f ${agenthome?}/bin/Agent.pid ]
    do
        sleep 1
    done
    agentpid=$(cat ${agenthome?}/bin/Agent.pid)
    [ -f ${agenthome?}/bin/LoggingAgent.pid ] && loggingpid=$(cat
${agenthome?}/bin/LoggingAgent.pid)
    echo "${agentpid?} ${loggingpid?}"
}

```

```

}

start() {
    ISFagentstatus ISFagents 2>&1 > ${agentlog?}
    RETVAL=$?
    echo -n "Starting ISFagents services: "
    if [ $RETVAL -ne 0 ]; then
        ${agentstart} > ${agentlog?} 2>> ${agentlog?}
        RETVAL=$?
        [ $RETVAL -eq 0 ] && touch /var/lock/subsys/ISFagents &&
agentpids > /var/run/ISFagents.pid
    fi
    retval
    echo
    echo_log
    return $RETVAL
}

stop() {
    ISFagentstatus ISFagents 2>&1 > ${agentlog?}
    RETVAL=$?
    echo -n "Stopping ISFagents services: "
    if [ $RETVAL -eq 0 ]; then
        ${agentstop} > ${agentlog?} 2>> ${agentlog?}
        RETVAL=$?
        [ $RETVAL -eq 0 ] && rm -f /var/lock/subsys/ISFagents && rm -rf
/var/run/ISFagents.pid
    fi
    retval
    echo
    echo_log
    return $RETVAL
}

ISFagentstatus() {
    local base=${1##*/}
    local pid

    echo -n "Getting ISFagents status: "
    if [ -f /var/run/${base}.pid ]; then
        local pidrc=0
        read pid < /var/run/${base}.pid
        if [ -n "$pid" ]; then
            for p in $pid

```

```

do
    ps -p $p -o pid 2>&1 > /dev/null
    let pidrc=pidrc+$?
done
RETVAL=${pidrc?}
if [ ${RETVAL?} -eq 0 ]; then
    echo "${base} (pid $pid) is running..."
    return ${RETVAL?}
fi
fi

status ISFagents
RETVAL=$?
return ${RETVAL?}
}

initialize
if [ "verbose" == "$1" ]; then
    VERBOSE=1
    shift
fi

# See how we were called.
case "$1" in
    start)
        start
        ;;
    stop)
        stop
        ;;
    status)
        ISFagentstatus ISFagents
        ;;
    restart)
        stop
        start
        ;;
    condrestart)
        if [ -f /var/lock/subsys/ISFagents ]; then
            stop
            start
        fi
        ;;
    *)

```

```

        echo $"Usage: $0 [verbose] {start|stop|status|restart|condrestart}"
        exit 1
    esac

    exit $RETVAL

```

Example: B-16 ISFserver.sh script

```

#!/bin/bash
#
# ISFserver      Startup script for ISFserver
# Written by    Anthony Corrente
#
# chkconfig: 5 98 2
# description: ISFserver is a service that is used by Information
Server \
#               to start/stop/query the status of, the
\
#               Information Services Framework server.
# pidfile: /var/run/ISFserver.pid

# Source function library.
. /etc/rc.d/init.d/functions

#
initialize() {
    VERBOSE=0
    RETVAL=0
    IS_INIT_D=true
    export IS_INIT_D
    wasbin=/opt/IBM/InformationServer/ASBServer/bin
    isfserver="${wasbin?}/MetadataServer.sh"
    washome=/opt/IBM/WebSphere/AppServer/profiles/default/logs/server1
    wasstart="${isfserver?} run"
    wasstop="${isfserver?} stop"
    wasproc=java
    waslog=/tmp/ISFserver.$$log
}

retval() {
    if [ $RETVAL -eq 0 ]; then
        echo_success
    else
        echo_failure
    fi
}

```



```

    fi
}

echo_log() {
    [ $VERBOSE -eq 1 ] && (cat ${waslog?}; rm -rf ${waslog?})
}

waspids(){
    while [ ! -f ${washome?}/server1.pid ]
    do
        sleep 1
    done
    waspid=$(cat ${washome?}/server1.pid)
    echo "${waspid?}"
}

start() {
    ISFserverstatus ISFserver 2>&1 > /dev/null
    RETVAL=$?
    echo -n "Starting ISFserver services: "
    if [ $RETVAL -ne 0 ]; then
        ${wasstart?} 2>&1 > ${waslog?}
        RETVAL=$?
        [ $RETVAL -eq 0 ] && touch /var/lock/subsys/ISFserver && waspids
    > /var/run/ISFserver.pid
    fi
    retval
    echo
    echo_log
    return $RETVAL
}

stop() {
    echo -n "Stopping ISFserver services: "
    ${wasstop?} 2>&1 > ${waslog?}
    RETVAL=$?
    [ $RETVAL -eq 0 ] && rm -f /var/lock/subsys/ISFserver && rm -rf
/var/run/ISFserver.pid
    retval
    echo
    echo_log
    return $RETVAL
}

ISFserverstatus() {

```

```

local base=${1##*/}
local pid

echo -n "Getting ISFserver status: "
if [ -f /var/run/${base}.pid ]; then
    local pidrc=0
    read pid < /var/run/${base}.pid
    if [ -n "$pid" ]; then
        for p in $pid
        do
            ps -p $p -o pid 2>&1 > /dev/null
            let pidrc=pidrc+$?
        done
        RETVAL=${pidrc?}
        if [ ${RETVAL?} -eq 0 ]; then
            echo "${base} (pid $pid) is running..."
            return ${RETVAL?}
        fi
    fi
    status ISFserver
    RETVAL=$?
    return ${RETVAL?}
}

initialize
if [ "verbose" == "$1" ]; then
    VERBOSE=1
    shift
fi

# See how we were called.
case "$1" in
    start)
        start
        ;;
    stop)
        stop
        ;;
    status)
        ISFserverstatus ISFserver
        ;;
    restart)
        stop

```

```

        start
        ;;
condrestart)
    if [ -f /var/lock/subsys/ISFserver ]; then
        stop
        start
    fi
    ;;
*)
    echo $"Usage: $0 [verbose] {start|stop|status|restart|condrestart}"
    exit 1
esac

exit $RETVAL

```

Example: B-17 IISGrid.sh script

```

#!/bin/bash
#
# IISGrid Startup script for IBM Information Server Grid
# Written by      Anthony Corrente
#
# chkconfig: 5 99 1
# description: This script is used to start/stop/query the status \
#              of the IBM Information Server (R) on Grid.

# Source function library.
. /etc/rc.d/init.d/functions

echo_running() {
    [ "$BOOTUP" = "color" ] && $MOVE_TO_COL
    echo -n "[ "
    [ "$BOOTUP" = "color" ] && $SETCOLOR_SUCCESS
    echo -n $"RUNNING"
    [ "$BOOTUP" = "color" ] && $SETCOLOR_NORMAL
    echo -n " ]"
    echo -ne "\r"
    return 0
}

echo_stopped() {
    [ "$BOOTUP" = "color" ] && $MOVE_TO_COL
    echo -n "[ "

```

```

[ "$BOOTUP" = "color" ] && $SETCOLOR_FAILURE
echo -n "$STOPPED"
[ "$BOOTUP" = "color" ] && $SETCOLOR_NORMAL
echo -n " ]"
echo -ne "\r"
return 0
}

REVAL=0

start() {
    RETVAL=0
    for service in xmeta ISFagents ISFserver dstage LoadL gmond
    do
        service ${service?} start
        let RETVAL=RETVAL+$?
    done
    [ $RETVAL -eq 0 ] && touch /var/lock/subsys/IISGrid
    return $RETVAL
}

stop () {
    RETVAL=0
    for service in gmond LoadL dstage ISFserver ISFagents xmeta
    do
        service ${service?} stop
        let RETVAL=RETVAL+$?
    done
    [ $RETVAL -eq 0 ] && rm -rf /var/lock/subsys/IISGrid
    return $RETVAL
}

echo_status () {
    if [ $RETVAL -eq 0 ]; then
        echo_running
    else
        echo_stopped
    fi
}

status () {
    iisRETVAL=0
    for service in xmeta ISFagents ISFserver dstage LoadL gmond
    do
        RETVAL=0

```

```

        servicelog=$(service ${service?} status)
        RETVAL=$?
        echo -n $"Status for ${service?} is"
        let iisRETVAL=iisRETVAL+RETVAL
        echo_status
        echo
        [ $VERBOSE -eq 1 ] && (echo ${servicelog}; echo)
    done
    echo -n $"Getting Information Server Grid status: "
    RETVAL=$iisRETVAL
    echo_status
    echo
    return $iisRETVAL
}

VERBOSE=0
if [ "verbose" == "$1" ]; then
    VERBOSE=1
    shift
fi

# See how we were called.

case "$1" in
    start)
        start
        ;;
    stop)
        stop
        ;;
    status)
        status
        ;;
    restart)
        stop
        start
        ;;
    condrestart)
        if [ -f /var/lock/subsys/IISGrid ]; then
            stop
            start
        fi
        ;;

```

```

*)
    echo $"Usage: $0 [verbose] {start|stop|status|restart|condrestart}"
    exit 1
esac

exit $RETVAL

```

Example: B-18 xmeta.sh script

```

#!/bin/bash
#
# xmeta          Startup script for xmeta
# Written by     Anthony Corrente
#
# chkconfig: 5 96 4
# description: xmeta is a service that is used by Information Server \
#              to start/stop/query the status of the xmeta server.
# pidfile: /var/run/xmeta.pid

# Source function library.
. /etc/rc.d/init.d/functions

#
initialize() {
    VERBOSE=0
    RETVAL=0
    xmetadb="DB2"
    if [ "$xmetadb" == "DB2" ]; then
        xmetauser=db2inst1
        xmetagrpid=$(id -g -n ${xmetauser?})
        xmetastart="db2start"
        xmetastop="db2stop force"
        xmetaproc="xmeta"
        xmetapid="db2nps 0 | grep db2sysc | awk '{print \$2}'"
        xmetalogs=/tmp/xmeta.$$log
        fixdb2cfg
    fi
}

fixdb2cfg() {
    # This function is used to deal with failing over DB2.
    # Checks if the current host name is in the in the current db2nodes.cfg
    # If it is not, it will modify the db2nodes.cfg to reflect the new
    hostname.
    host=`hostname`

```

```

db2home=/home/${xmetauser?}/sqllib
db2nodes=${db2home?}/db2nodes.cfg
if ! grep ${host?} ${db2nodes?} 2>&1 > /dev/null; then
    line=$(cat ${db2nodes?})
    echo ${line?} | sed 's/ [a-zA-Z0-9.]* / '${host?}' /' >
${db2nodes?}
    chown ${xmetauser?}.${xmetagr?} ${db2nodes?}
    chmod "ug=rw,o=r" ${db2nodes?}
fi
}

xmetapid() {
    su -l ${xmetauser?} -c "${xmetapid?}"
}

retval() {
    if [ $RETVAL -eq 0 ]; then
        echo_success
    else
        echo_failure
    fi
}

echo_log() {
    [ $VERBOSE -eq 1 ] && (cat ${xmetalog?}; rm -rf ${xmetalog?})
}

start() {
    xmetastatus xmeta 2>&1 > ${xmetalog?}
    RETVAL=$?
    echo -n "Starting xmeta services: "
    if [ $RETVAL -ne 0 ]; then
        su -l ${xmetauser?} -c "${xmetastart?}" 2>&1 > ${xmetalog?}
        RETVAL=$?
        [ $RETVAL -eq 0 ] && touch /var/lock/subsys/xmeta && xmetapid >
/var/run/xmeta.pid
    fi
    retval
    echo
    echo_log
    return $RETVAL
}

stop() {
    xmetastatus xmeta 2>&1 > ${xmetalog?}

```

```

RETVAL=$?
echo -n "Stopping xmeta services: "
if [ $RETVAL -eq 0 ]; then
    su -l ${xmetauser?} -c "${xmetastop?}" 2>&1 > ${xmetalog?}
    RETVAL=$?
    [ $RETVAL -eq 0 ] && rm -f /var/lock/subsys/xmeta && rm -rf
/var/run/xmeta.pid
    fi
    retval
    echo
    echo_log
    return $RETVAL
}

xmetastatus() {
    local base=${1##*/}
    local pid

    echo -n "Getting xmeta status: "
    if [ -f /var/run/${base}.pid ]; then
        local pidrc=0
        read pid < /var/run/${base}.pid
        if [ -n "$pid" ]; then
            for p in $pid
            do
                ps -p $p -o pid 2>&1 > /dev/null
                let pidrc=pidrc+$?
            done
            RETVAL=${pidrc?}
            if [ $RETVAL -eq 0 ]; then
                echo "${base} (pid $pid) is running..."
                return $RETVAL
            fi
        fi
    fi
    status xmeta
    RETVAL=$?
    return $RETVAL
}

initialize
if [ "verbose" == "$1" ]; then
    VERBOSE=1
    shift
fi

```



```

# See how we were called.
case "$1" in
    start)
        start
        ;;
    stop)
        stop
        ;;
    status)
        xmetastatus xmeta
        ;;
    restart)
        stop
        start
        ;;
    condrestart)
        if [ -f /var/lock/subsys/xmeta ]; then
            stop
            start
        fi
        ;;
    *)
        echo $"Usage: $0 [verbose] {start|stop|status|restart|condrestart}"
        exit 1
esac

exit $RETVAL

```

Example: B-19 Loadl.sh script

```

#!/bin/sh
#
# LoadL:Starts / Stops the IBM LoadLeveler product
#
# chkconfig: 345 98 2
# description: Start or Stop IBM LoadLeveler
#
# processname: LoadL_master
# pidfile: /var/run/LoadL.pid
#

# Source function library.
. /etc/rc.d/init.d/functions

```

```

initialize() {
    RETVAL=0
    VERBOSE=0
    lluser=loadl
    llstart="llctl start"
    llstop="llctl stop"
    llreload="llctl reconfig"
    llstatus="llstatus"
    llproc="/opt/ibmll/LoadL/full/bin/LoadL_master"
    lllog=/tmp/LoadL.$$log
}

echo_status() {
    if [ $RETVAL -eq 0 ]; then
        echo_success
    else
        echo_failure
    fi
}

echo_log() {
    [ $VERBOSE -eq 1 ] && (cat ${lllog?}; rm ${lllog?})
}

llpid() {
    pidof ${llproc?}
}

start() {
    echo -n "Starting IBM LoadLeveler: "
    su -l ${lluser?} -c "${llstart?}" 2>&1 > ${lllog?}
    RETVAL=$?
    echo_status
    [ $RETVAL -eq 0 ] && touch /var/lock/subsys/LoadL && llpid >
/var/run/LoadL.pid
    echo
    echo_log
}

stop() {
    echo -n "Stopping IBM LoadLeveler: "
    su -l ${lluser?} -c "${llstop?}" 2>&1 > ${lllog?}
    RETVAL=$?
    echo_status
}

```

```

        [ $RETVAL -eq 0 ] && rm -rf /var/lock/subsys/LoadL && rm -rf
/var/run/LoadL.pid
        echo
        echo_log
    }

reload() {
    echo -n "Reloading IBM LoadLeveler Configuration: "
    su -l ${lluser?} -c "${llreload?}" 2>&1 > ${lllog?}
    RETVAL=$?
    echo_status
    echo
    echo_log
}

initialize
if [ "verbose" == "$1" ]; then
    VERBOSE=1
    shift
fi

# See how we were called.
case "$1" in
    start)
        start
        ;;
    stop)
        stop
        ;;
    status)
        status ${llproc?}
        RETVAL=$?
        [ $VERBOSE -eq 1 ] && su -l ${lluser?} -c "${llstatus?}" 2>&1
        ;;
    restart)
        stop
        sleep 1
        start
        ;;
    condrestart)
        if [ -f /var/lock/subsys/LoadL ]; then
            stop
            start
        fi
        ;;

```

```

        reload|reconfig)
if [ -f /var/lock/subsys/LoadL ]; then
    reload
fi
;;
*)
    echo $"Usage: $0 [verbose]
{start|stop|status|restart|condrestart|reload|reconfig}"
;;
esac

[ -f ${llog?} ] && rm -rf ${llog?}
exit $RETVAL

```

Example: B-20 install.sh script

```

#!/bin/bash
# Install services
# Written for installing Information Server for Grid services.
# Written by      Anthony Corrente

INSTALLDIR=.
INITDIR=/etc/rc.d/init.d

for service in dstage xmeta ISFagents ISFserver IIServer IISGrid
do
    cp ${INSTALLDIR?}/${service?} ${INITDIR?}/${service?}
    chkconfig --add ${service?}
    chkconfig ${service?} off
    chkconfig --list ${service?}
done

# Uncomment the following 3 lines to have IISGrid service start on
#reboot.
#service=IISGrid
#chkconfig --level 5 ${service?} on
#chkconfig --list ${service?}

```

Example: B-21 IIServer output examples

```
1) # service IIServer
Usage: /etc/init.d/IIServer [verbose]
{start|stop|status|restart|reload|condrestart}

2) # service IIServer status
Status for xmeta is [ RUNNING ]
Status for ISFagents is [ RUNNING ]
Status for ISFserver is [ RUNNING ]
Status for dstage is [ RUNNING ]
Getting Information Server status: [ RUNNING ]

3) # service IIServer verbose status
Status for xmeta is [ RUNNING ]
Getting xmeta status: xmeta (pid 32232) is running...

Status for ISFagents is [ RUNNING ]
Getting ISFagents status: ISFagents (pid 27327 27295) is running...

Status for ISFserver is [ RUNNING ]
Getting ISFserver status: ISFserver (pid 27599) is running...

Status for dstage is [ RUNNING ]
Getting dstage status: dsrpcd (pid 30814) is running...

Getting Information Server status: [ RUNNING ]

4) # service IIServer verbose stop
Stopping dstage services: [ OK ]
Stopping ISFserver services: [ OK ]
Stopping ISFagents services: [ OK ]
Stopping xmeta services: [ OK ]

5) # service IIServer verbose status
Status for xmeta is [ STOPPED ]
Getting xmeta status: xmeta is stopped

Status for ISFagents is [ STOPPED ]
Getting ISFagents status: ISFagents is stopped

Status for ISFserver is [ STOPPED ]
Getting ISFserver status: ISFserver is stopped

Status for dstage is [ STOPPED ]
Getting dstage status: dsrpcd is stopped

Getting Information Server status: [ STOPPED ]
```

```

6) # service IIServer status
Status for xmeta is          [ STOPPED ]
Status for ISFagents is      [ STOPPED ]
Status for ISFserver is      [ STOPPED ]
Status for dstage is         [ STOPPED ]
Getting Information Server status: [ STOPPED ]

7) # service IIServer verbose start
Starting xmeta services:     [ OK ]
Starting ISFagents services: [ OK ]
Starting ISFserver services: [ OK ]*
Starting dstage services:    [ OK ]

* this can take a longer than expected time to return OK

8) should a command be interrupted it may be restarted using
   # service IIServer start
Starting xmeta services:     [FAILED]
Starting ISFagents services: [ OK ]
Starting ISFserver services: [FAILED]
Starting dstage services:    [ OK ]

9) Check status again:
   # service IIServer status
Status for xmeta is          [ RUNNING ]
Status for ISFagents is      [ RUNNING ]
Status for ISFserver is      [ RUNNING ]
Status for dstage is         [ RUNNING ]
Getting Information Server status: [ RUNNING ]
]

```

Example: B-22 uninstall.sh script

```

#!/bin/bash
# Uninstall services
# Written for removing Information Server for Grid services.
# Written by      Anthony Corrente

INITDIR=/etc/rc.d/init.d/

for service in dstage xmeta IISagents WAServer IIServer IISGrid
do
    chkconfig ${service?} off
    chkconfig --del ${service?}
    chkconfig --list ${service?}
    rm ${INITDIR?}/${service?}
done

```

B.5.3 HA scripts

We developed the following HA scripts:

► Failover to NILE scripts

These scripts support failover from the original front end node (orion) to the standby node (nile):

- failover_Nile.sh
- mountsanfs_Nile.sh
- remounts_Nile.sh

These scripts are described in “Failover to NILE scripts” on page 289.

► Failback to ORION scripts

These scripts support failback from the standby node (nile) to the original front end node (orion):

- failback_Nile.sh
- umountsanfs_Nile.sh
- failback_Orion.sh
- mountsanfs_Orion.sh
- remounts_Orion.sh

These scripts are described in “Failback to ORION scripts” on page 294.

B.5.3.1 Failover to NILE scripts

Scripts to support failover from the original front end node (orion) to the standby node (nile) include failover_Nile.sh, mountsanfs_nile.sh, and remount_Nile.sh as follows:

- failover_Nile.sh (shown in Example B-23) transfers the front-end role to the standby node (nile). It invokes a number of scripts such as mountsanfs_Nile.sh (shown in Example B-24 on page 292) and the services scripts identified earlier.

This script executes on the standby node (nile).

Example: B-23 failover_Nile.sh script

```
#!/bin/bash
#
# failover_Nile -- Set up Nile to be the primary head node.
#
# Written by      Anthony Corrente
# Reviewed by    Nagraj Alur, and Sachiko Toratani
#
# Description:
```

```

# This script can be used to set Nile to be the primary head node.
# When set to primary, the following requirements need to be set and/or
met:
# 1) Virtual IP up
# 2) mount the SAN filesystems
# 3) Start Ganglia, including httpd service
# 4) Loadleveler configuration set to the correct central manager
# 5) NFS server setup
# 6) Start IISGrid service
#

init() {
    lluser="loadl"
    llhome=/home/loadl
    hostname=`hostname -s` ;# short hostname
    virtualhostname="kazan.itsosj.sanjose.ibm.com"
}

primary() {
    ## Set up as primary:
    ## Setup LoadLeveler
    echo $"Setting current host as primary headnode..."
    virtualIPifup
    mountsanfsNile
    setlladmin
    startnfsserver
    startganglia
    startIISGrid
    echo $"The head node should now be up. Please check."
    echo $"Please run umount_Nile.sh to remount the NFS mounts on the
compute nodes defined in the nodes.cfg file."
}

virtualIPifup() {
    ## up the virtual IP addresses
    echo $"Upping Virtual IP Addresses"
    ifup eth0:1 && ifconfig eth0:1
    ifup eth1:1 && ifconfig eth1:1
}

mountsanfsNile () {
    ## This will run the mountsanfs_Nile.sh script located under /root/
    echo $"Mounting SAN filesystems"
    /root/mountsanfs_Nile.sh
    df -k
}

```



```

}

setlladmin () {
echo $"Setting LoadL_admin configuration file central_manager to
${hostname?}"
## Setup LoadLeveler
ex ${llhome?}/LoadL_admin <<LLADMIN
/central_manager = true
?^[a-z]*:
s/^[a-z]*:/${hostname?}:
wq!
LLADMIN
}

startnfsserver() {
## Restart nfs server
echo $"Restarting NFS server"
exportfs -a 2>&1 > /dev/null
service nfs restart
exportfs -a 2>&1 > /dev/null
}

startganglia() {
## Start Ganglia service
if service gmetad status; then
    echo $"Ganglia gmetad, gmond, and httpd services are already
running"
else
    echo $"Starting Ganglia gmetad, gmond, and httpd services..."
    service gmetad start
    service gmond start
    service httpd start
fi
}

startIISGrid(){
## This will start the IISGrid services
service IISGrid start
}

checkvirtualIPisup() {
# returns 0 if virtual host is still up
#      1 if not up.
    echo $"Checking if virtual IP addresses are still up...(this takes
approximately 3 seconds.)"

```

```

        ping -c 3 -i 1 ${virtualhostname?} 2>&1 > /dev/null
        return $?
    }

    main() {
        if ! checkvirtualIPisup; then
            primary
        else
            echo $"Virtual IP addresses still running."
            echo $"Please re-run this script when head node is down."
        fi
    }

    init
    main $@

```

-
- `mountsanfs_Nile.sh` (shown in Example B-24) mounts the file systems on the shared SAN volumes on the standby node (nile). It is invoked by the `failover_Nile.sh` script.

Example: B-24 mountsanfs_Nile.sh script

```

#!/bin/bash
#
# mountsanfs_Nile
# This script mounts all the SAN filesystems to
# /home, /opt/IBM, /ds_overview, /opt/mqm
#
# /dev/sdd1 196870980 32122304 154748216 18% /home
# /dev/sdd2 98435476 13068052 80367196 14% /opt/IBM
# /dev/sdd3 98435476 95684 93339564 1% /ds_overview
# /dev/sdd5 71157244 554568 66988056 1% /opt/mqm

function fmount {
    # This function takes 2 arguments
    # dev - the device
    # mtp - the mount point
    dev=$1
    mtp=$2
    if [ -b $dev -a -d $mtp ]
    then
        mount $dev $mtp || echo "failed to mount $dev to $mtp"
    fi
}

```

```
fmount /dev/sdd1 /home
fmount /dev/sdd2 /opt/IBM
fmount /dev/sdd3 /ds_overview
fmount /dev/sdd5 /opt/mqm
```

- remounts_Nile.sh (shown in Example B-25) mounts the SAN file systems mounted on nile by the mountsanfs_Nile.sh script on each of the compute nodes.

It executes on the standby node (nile).

Example: B-25 remount_Nile.sh script

```
#!/bin/bash
#
# remount_Nile
# This script kills the resource tracker and unmount/mount the
# filesystems on the compute nodes

cat nodes.cfg
echo "-----"
echo "kill resource tracker on the above compute nodes"
./runcmd "kill -9 \$(ls -l /dev/null | grep '/opt/IBM' | grep
'resour' | awk '{print \$2}' | sort | uniq)"
echo "-----"
echo "run umount -a"
./runcmd "umount -a"
echo "-----"
echo "run df"
./runcmd "df"
echo "-----"
echo "sleep for 3 minutes"
sleep 3m
echo "-----"
echo "run mount -a"
./runcmd "mount -a"
echo "-----"
echo "run df"
./runcmd "df"
```

B.5.3.2 Failback to ORION scripts

Scripts to support failback from the standby node (nile) to the original front end node (orion) such as failback_Nile.sh, umountsanfs_nile.sh, failback_Orion.sh, mountsanfs_Orion.sh, and remount_Orion.sh as follows:

- failback_Nile.sh (shown in Example B-26) performs an orderly shutdown of the front end node role on nile, and restores it as a standby node. It invokes other scripts such as umountsanfs_Nile.sh and services scripts identified earlier.

This script executes on the standby node (nile).

Example: B-26 failback_Nile.sh script

```
#!/bin/bash
#
# failback_Nile -- Set up Nile as standby head node.
#
# Written by      Anthony Corrente
# Reviewed by    Nagraj Alur, and Sachiko Toratani
#

# Description:
# This script can be used to set Nile as the standby head node.
# When set to standby, the following requirements need to be set:
# 1) Stop IISGrid service
# 2) Shutdown Ganglia and httpd services
# 3) NFS server switched off
# 4) Unmount the SAN filesystems
# 5) Virtual IP down.
#

init() {
    lluser="load1"
    llhome="/home/load1"
    hostname=`hostname -s` ;# short hostname
    virtualhostname="kazan.itsosj.sanjose.ibm.com"
}

standby() {
    ## reverse what we do for primary
    echo $"Setting current host as standby headnode..."
    stopIISGrid
    stopganglia
    stopnfsserver
    umountsanfsNile
}
```

```

        virtualIPifdown
    }

stopIISGrid(){
## This will stop the IISGrid services
    service IISGrid stop
}

stopganglia() {
## Stop Ganglia service
if service gmetad status; then
    echo $"Stopping Ganglia gmetad, gmond, and httpd services..."
    service gmetad stop
    service gmond stop
    service httpd stop
else
    echo $"Ganglia gmetad, gmond, and httpd services are already
stopped."
fi
}

stopnfsserver() {
## Stop nfs server
echo $"Stopping NFS server"
service nfs stop
}

umountsanfsNile() {
## This will run the umountsanf_Nile.sh script located under /root/
    echo $"Unmounting SAN filesystems"
    /root/umountsanfs_Nile.sh
    df -k
}

virtualIPifdown() {
## down the virtual IP addresses
    echo $"Downing Virtual IP Addresses"
    ifdown eth0:1 && ifconfig eth0:1
    ifdown eth1:1 && ifconfig eth1:1
}

checkvirtualIPisup() {
# returns 0 if virtual host is still up
#      1 if not up.

```

```

        echo $"Checking if virtual IP addresses are still up...(this takes
approximately 3 seconds.)"
        ping -c 3 -i 1 ${virtualhostname?} 2>&1 > /dev/null
        return $?
    }

    checkvirtualIPislocal () {
        ## check to see if the virtual IP address is running locally.
        # returns 0 if virtual IP address is running locally.
        #      1 if virtual IP address is not running locally.
        ifconfig eth0:1 | grep inet
        return $?
    }

    main() {
        if checkvirtualIPisup && checkvirtualIPislocal ; then
            standby
        else
            echo $"Virtual IP addresses are not up."
            echo $"This may already be a standby node OR"
            echo $"is not the current primary head node"
        fi
    }

    init
    main $@

```

- `umountsanfs_Nile.sh` (shown in Example B-27) performs an unmount of all the file systems on the shared SAN drive. It is invoked by the `failback_Nile.sh` script.

Example: B-27 `umountsanfs_Nile.sh` script

```

#!/bin/bash
#
# umountsanfs_Nile
# This script ummounts all the SAN filesystems from
# /home,/opt/IBM, /ds_overview, /opt/mqm
#
if lsof | egrep '/home/|/opt/IBM/|/ds_overview|/opt/mqm'
then
    echo $"Please review the above output and kill pids"
else
    umount /home
    umount /opt/IBM

```

```
    umount /ds_overview
    umount /opt/mqm
fi
```

- failback_Orion.sh (shown in Example B-28) restores the original front end node (orion) to its front end node role. It invokes other scripts such as mountsanfs_Orion.sh and the services scripts identified earlier.

This script executes on the original front end node (orion).

Example: B-28 failback_Orion.sh script

```
#!/bin/bash
#
# failback_Orion -- Set up Orion to be the primary head node.
#
# Written by      Anthony Corrente
# Reviewed by    Nagraj Alur, and Sachiko Toratani
#
# Description:
# This script can be used to set Orion to be the primary head node.
# When set to primary, the following requirements need to be set and/or
# met:
# 1) Virtual IP up
# 2) mount the SAN filesystems
# 3) Start Ganglia, including httpd service
# 4) Loadleveler configuration set to the correct central manager
# 5) NFS server setup
# 6) Start IISGrid service
#
init() {
    lluser="loadl"
    llhome=/home/loadl
    hostname=`hostname -s` ;# short hostname
    virtualhostname="kazan.itsosj.sanjose.ibm.com"
}

primary() {
## Set up as primary:
## Setup LoadLeveler
echo $"Setting current host as primary headnode..."
virtualIPifup
mountsanfsOrion
setlladmin
```

```

        startnfsserver
        startganglia
        startIISGrid
        echo $"The head node should now be up. Please check."
        echo $"Please run remount_Orion.sh to remount the NFS mounts on the
compute nodes defined in the nodes.cfg file."
    }

    virtualIPifup() {
    ## up the virtual IP addresses
        echo $"Upping Virtual IP Addresses"
        ifup eth0:1 && ifconfig eth0:1
        ifup eth1:1 && ifconfig eth1:1
    }

    mountsanfsOrion () {
    ## This will run the mountsanfs_Orion.sh script located under /root/
        echo $"Mounting SAN filesystems"
        /root/mountsanfs_Orion.sh
        df -k
    }

    setlladmin () {
    echo $"Setting LoadL_admin configuration file central_manager to
    ${hostname?}"
    ## Setup LoadLeveler
    ex ${llhome?}/LoadL_admin <<LLADMIN
    /central_manager = true
    ?^[a-z]*:
    s/^[a-z]*:/${hostname?}:
    wq!
    LLADMIN
    }

    startnfsserver() {
    ## Restart nfs server
        echo $"Restarting NFS server"
        exportfs -a 2>&1 > /dev/null
        service nfs restart
        exportfs -a 2>&1 > /dev/null
    }

    startganglia() {
    ## Start Ganglia service
        if service gmetad status; then

```



```

        echo $"Ganglia gmetad, gmond, and httpd services are already
running"
    else
        echo $"Starting Ganglia gmetad, gmond, and httpd services..."
        service gmetad start
        service gmond start
        service httpd start
    fi
}

startIISGrid(){
## This will start the IISGrid services
    service IISGrid start
}

checkvirtualIPisup() {
# returns 0 if virtual host is still up
#     1 if not up.
    echo $"Checking if virtual IP addresses are still up...(this takes
approximately 3 seconds.)"
    ping -c 3 -i 1 ${virtualhostname?} 2>&1 > /dev/null
    return $?
}

main() {
    if ! checkvirtualIPisup; then
        primary
    else
        echo $"Virtual IP addresses still running."
        echo $"Please re-run this script when head node is down."
    fi
}

init
main $@

```

- mountsanfs_Orion.sh (shown in Example B-29) mounts the file systems on the shared SAN volumes on the original front end node (orion).

This script executes on the original front end node (orion).

Example: B-29 mountsanfs_Orion.sh script

```
#!/bin/bash
#
# mountsanfs
# This script mounts all the SAN filesystems to
# /home, /ds_overview, /data, /opt/mqm, and /opt/IBM.
#
# /dev/sdf1 196870980 32122304 154748216 18% /home
# /dev/sdf2 98435476 13068052 80367196 14% /opt/IBM
# /dev/sdf3 98435476 95684 93339564 1% /ds_overview
# /dev/sdf5 71157244 554568 66988056 1% /opt/mqm

function fmount {
# This function takes 2 arguments
# dev - the device
# mtp - the mount point
dev=$1
mtp=$2
if [ -b $dev -a -d $mtp ]
then
mount $dev $mtp || echo "failed to mount $dev to $mtp"
fi
}
fmount /dev/sdf1 /home
fmount /dev/sdf2 /opt/IBM
fmount /dev/sdf3 /ds_overview
fmount /dev/sdf5 /opt/mqm
```

- remounts_Orion.sh (shown in Example B-30) mounts the SAN file systems mounted on orion by the mountsanfs_Orion.sh script on each of the compute nodes.

It executes on the original front end node (orion).

Example: B-30 remount_Orion.sh script

```
#!/bin/bash
#
# remount_Orion
# This script kills the resource tracker and unmount/mount the
# filesystems on the compute nodes

cat nodes.cfg
echo "-----"
echo "kill resource tracker on the above compute nodes"
./runcmd "kill -9 \$(lsof 2> /dev/null | grep '/opt/IBM' | grep
'resour' | awk '{print \$2}' | sort | uniq)"
echo "-----"
echo "run umount -a"
./runcmd "umount -a"
echo "-----"
echo "run df"
./runcmd "df"
echo "-----"
echo "sleep for 3 minutes"
sleep 3m
echo "-----"
echo "run mount -a"
./runcmd "mount -a"
echo "-----"
echo "run df"
./runcmd "df"
```

Parallel Framework

In this appendix we provides a brief overview of the Parallel Framework.

C.1 Parallel Framework

Jobs developed with DataStage Enterprise Edition and QualityStage are independent of the actual hardware and degree of parallelism used to run the job. The parallel configuration file defines logical processing nodes that provides a runtime mapping between the job and the actual infrastructure and resources used at execution.

To facilitate scalability across the boundaries of a single server, and to maintain platform independence, the parallel framework uses a multi-process architecture.

The runtime architecture of the parallel framework uses a process-based architecture that enables scalability beyond server boundaries while avoiding platform-dependent threading calls. The actual runtime deployment for a given job design is composed of a hierarchical relationship of operating system processes, running on one or more physical servers as shown in Figure C-1.

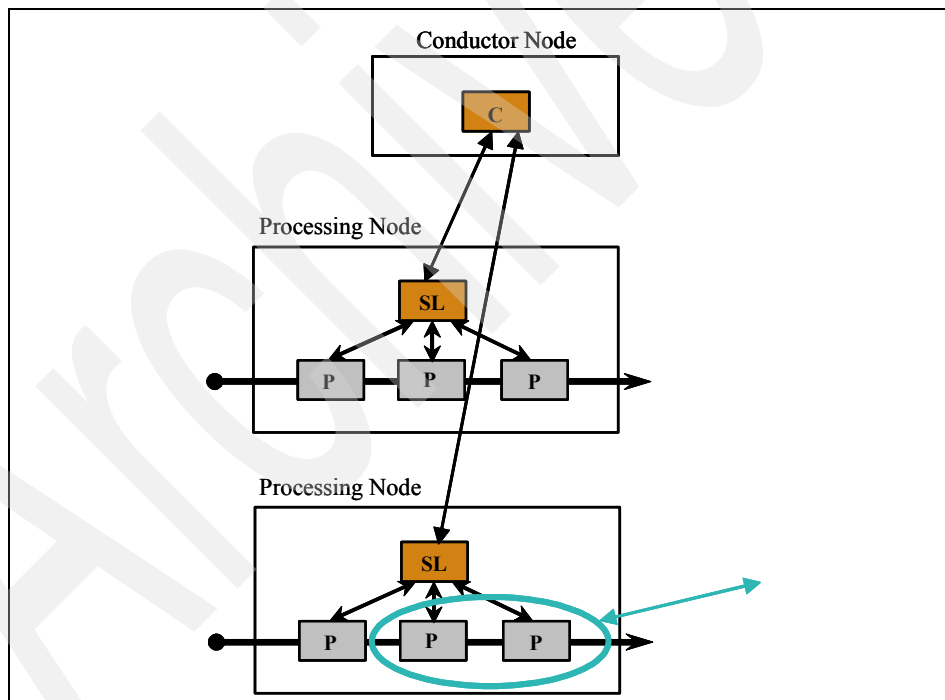


Figure C-1 Process hierarchy of the parallel framework

Each of these processes is briefly described here as follows:

- ▶ Conductor Node (one per job) is the main process used to start jobs, determine resource assignments, and create Section Leader (SL) processes on one or more processing nodes. It acts as a single coordinator for status and error messages, and manages orderly shutdown when processing completes or in the event of a fatal error. The conductor node is run from the primary server.
- ▶ SLs (one per logical processing node) are used to create and manage player processes which perform the actual job execution. The SLs also manage communication between the individual player processes and the master Conductor Node.
- ▶ Players are one or more logical groups of processes used to execute the data flow logic. All players are created as groups on the same server as their managing SL process.

The parallel framework uses several different channels for interprocess communication as shown in Figure C-2 on page 306. Within the same server (such as a compute node), the communication channel is established through shared memory, although this can be changed using environment variable settings. When communication is across servers, such as grid implementations with multiple compute nodes, TCP network protocol is used. The other channels are as follows:

- ▶ control channel is used for sending control messages to/from the conductor to section leaders and section leaders to/from processes
- ▶ stdout channel/pipe supports status messages
- ▶ stderr channel/pipe supports error status codes and messages
- ▶ APT_Communicator is used to send data directly from upstream to downstream player processes. These paths support high data volumes and cross-machine communication must be minimized. The data flow is directly between player processes to maximize scalability. The Conductor and Section Leaders do not pass data, but rather facilitate management, status, and error messages.

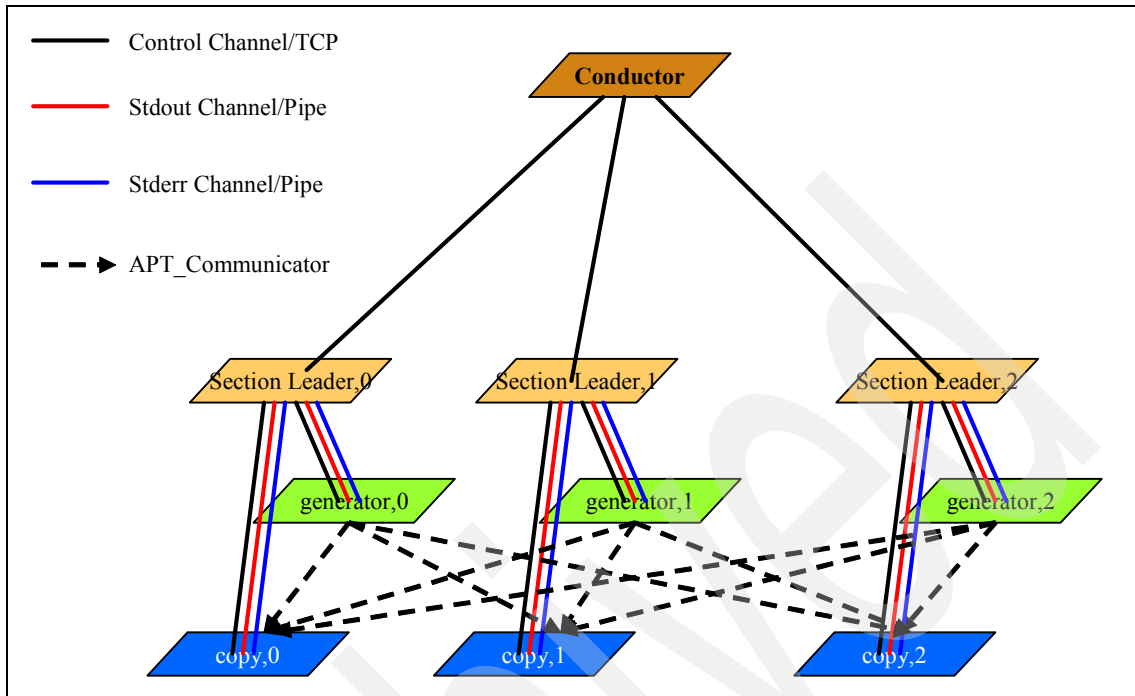


Figure C-2 Runtime control and data networks

Additional material

This book refers to additional material that can be downloaded from the Internet as described below.

D.1 Locating the Web material

The Web material associated with this book is available in softcopy on the Internet from the IBM Redbooks Web server. Point your Web browser at:

<ftp://www.redbooks.ibm.com/redbooks/SG247625>

Alternatively, you can go to the IBM Redbooks Web site:

ibm.com/redbooks

Select the **Additional materials** and open the directory that corresponds with the IBM Redbooks form number, SG247625.

D.2 Using the Web material

The additional Web material that accompanies this book includes the SG247625.zip file, a collection of compressedscript examples

D.2.1 System requirements for downloading the Web material

The following system configuration is recommended:

Hard disk space:	100 KB
Operating System:	RedHat Enterprise Linux

D.2.2 How to use the Web material

Create a subdirectory (folder) on your workstation, and unzip the contents of the Web material zip file into this folder.

Related publications

The publications listed in this section are considered suitable for a more detailed discussion of the topics covered in this book.

IBM Redbooks

For information about ordering these publications, see “How to get Redbooks” on page 310. Note that some of the documents referenced here may be available in softcopy only.

- ▶ *SOA Solutions Using IBM Information Server*, SG24-7402
- ▶ *IBM WebSphere Information Analyzer and Data Quality Assessment*, SG24-7508
- ▶ *IBM WebSphere QualityStage Methodologies, Standardization, and Matching*, SG24-7546
- ▶ *IBM WebSphere QualityStage Methodologies, Standardization, and Matching*, SG24-7546

Other publications

These publications are also relevant as further information sources:

- ▶ *IBM Information Server - Delivering information you can trust*, IBM United States Announcement 206-308 dated December 12, 2006
- ▶ *Information Server Planning, Installation, and Configuration Guide Version 8.0.1*, GC19-1048-01
- ▶ *Information Server Introduction Version 8.0.1*, SC19-1049-01
- ▶ *Information Server Administration Guide Version 8.0.1*, SC18-9929
- ▶ *IBM Information Server Reporting Guide Version 8.0.1*, SC19-1162-01
- ▶ *IBM Information Server Quick Start Guide*
- ▶ *IBM Information Server—Delivers next generation data profiling analysis and monitoring through the new IBM WebSphere Information Analyzer module*, IBM United States Announcement 207-043 dated March 13th 2007

- ▶ *IBM Information Management Software Profiling: Take the first step toward assuring data quality*, December 2006, IMW11808-USEN
- ▶ *IBM WebSphere DataStage Parallel Job Developer Guide*, LC18-9891
- ▶ *IBM WebSphere DataStage Parallel Job Advanced Developer Guide Version 8.0.1*, LC18-9892-01
- ▶ *IBM WebSphere DataStage Designer Client Guide*, LC18-9893
- ▶ *IBM WebSphere DataStage Director Client Guide*, LC18-9894
- ▶ *IBM WebSphere DataStage Administrator Client Guide*, LC18-9895
- ▶ *IBM WebSphere DataStage BASIC Reference Guide*, LC18-9897
- ▶ *IBM WebSphere DataStage Server Job Developer Guide*, LC18-9898
- ▶ *IBM WebSphere DataStage Parallel Engine Message Reference Version 8*, LC18-9931
- ▶ *IBM WebSphere DataStage Connectivity Guide for IBM DB2 Databases*, LC18-9932

Online resources

These Web sites are also relevant as further information sources:

- ▶ IBM Information Server information center
<http://publib.boulder.ibm.com/infocenter/iisinfsv/v8r0/index.jsp>
- ▶ Installing Linux on IBM systems
<http://www.ibm.com/developerworks/systems/library/es-networkinstall/index.html>

How to get Redbooks

You can search for, view, or download Redbooks, Redpapers, Technotes, draft publications and Additional materials, as well as order hardcopy Redbooks, at this Web site:

ibm.com/redbooks

Help from IBM

IBM Support and downloads

ibm.com/support

IBM Global Services

ibm.com/services

Archived

Index

Symbols

#\$APT_GRID_SEQFILE_HOST# 176
\$APT_CONFIG_FILE 7, 12, 14, 19, 21, 39, 176
\$APT_GRID_COMPUTENODES 17–18, 140, 148, 167, 187
\$APT_GRID_CONFIG 140
\$APT_GRID_ENABL 18
\$APT_GRID_ENABLE 17, 21–22, 24, 140, 148, 159, 167, 187
\$APT_GRID_IDENTIFIER 141
\$APT_GRID_OPTS 141
\$APT_GRID_PARTITIONS 17–18, 24, 141, 148, 159, 167, 187
\$APT_GRID_QUEUE 141
\$APT_GRID_SCRIPTPOST 141
\$APT_GRID_SCRIPTPRE 142
\$APT_GRID_SEQFILE_HOST 19, 21–22, 142, 149, 167–168, 187
\$APT_GRID_SEQFILE_HOST2 19, 21–22, 142, 149, 167–168
\$APT_GRID_STAT_CMD 142
\$APT_PM_CONDUCTOR_HOSTNAME 14, 74, 135
\$DS_HOSTNAME_ALIAS 66

A

A2Z Financial Services Inc.
 business requirement 48
add and remove nodes 217
ailback_Orion.sh script 289, 297
alias 69
anaconda-ks.cfg file 109

B

best practices for the Information Server grid environment
 configuration guidelines 40
 compute nodes 41
 front end node 40
 network 40
 tuning guidelines 42
boot sequence 145

build a grid environment using the BYOG toolkit
 prepare the PXE boot server on the front end node 86
 add the compute nodes to Loadl_admin file 108
 create RedHat installation tar image and copy it to the /tftpboot directory 88
 create the kickstart file for each new compute node 109
 install and configure a DHCP service 93
 install and configure a TFTP service 96
 install BYOG tar file under the /tftpboot directory 91
 modify the post-env file 93
 verify /tftpboot directory structure 92
 PXEBoot execution on a compute node 116
 set up NFS, NIS, Workload Scheduler, SSH, and Ganglia on the front end node 78
 Ganglia 97
 NFS 78
 NIS 80
 SSH 85
 Workload Scheduler LoadLeveler 102
Build Your Own Grid (BYOG) toolkit 45, 93, 131, 244

C

cancel a waiting job 214
canjob command 258
central_manager 108
classes 128
clustered scenario 27
commands and scripts 252
compute node 9–10, 14, 18, 23, 39, 51, 68, 74, 108, 128
conductor node 9, 74, 135, 305
configuration file 25
CPUs 40
create_ks.sh script 113
cron utility 85, 145

D

- default.apt 176
 - configuration file 75
- device names 223
- DHCP (Dynamic Host Configuration Protocol) 86, 116
 - clients 94
 - server 120, 123
 - service 93, 96
- dhcpd file 94
- dhcpd.conf file 94, 112, 120, 130
- disk storage 41
- domain name 71
- dsadm 131
- dsenv 17, 67, 131, 134–136
- DSParams 147
 - file 18
- dstage.sh script 268
- dynamic configuration file 31, 186, 247
- Dynamic_Grid.sh script 24–25
- Dynamic_grid.sh script 19, 142

E

- enterprise grid computing 4
- environment variables 19, 38, 66, 74, 93, 135, 139, 168
 - \$APT_GRID_COMPUTENODES 140
 - \$APT_GRID_CONFIG 140
 - \$APT_GRID_ENABLE 140
 - \$APT_GRID_IDENTIFIER 141
 - \$APT_GRID_OPTS 141
 - \$APT_GRID_PARTITIONS 141
 - \$APT_GRID_QUEUE 141
 - \$APT_GRID_SCRIPTPOST 141
 - \$APT_GRID_SCRIPTPRE 142
 - \$APT_GRID_SEQFILE_HOST 142
 - \$APT_GRID_SEQFILE_HOST2 142
 - \$APT_GRID_STAT_CMD 142
 - grid related 139
- Ethernet switch 40
- execute command activity stage 20–21, 187
- exportfs command 79
- exports file 79, 144

F

- failback 46, 236
 - failback_Nile.sh script 236, 294
 - failback_Orion.sh 289, 297

- failback_Orion.sh script 237
- failover 46, 221
 - failover_Nile.sh script 225, 237, 289
- fastname 135
- financial services business scenario
 - business requirement 48
 - environment configuration 49
 - general approach 52
- front end node 9, 11–12, 16, 39, 49, 61, 68, 98, 123, 229

G

- Ganglia 97
 - ganglia_gmond script 93, 124
 - services 101
- gateway IP address 61
- GFS (Global FileSystem) 13–14
- global file system 13
- grid computing execution flow 24
 - grid computing vs. cluster computing 27
 - grid enable enterprise applications 18
 - execution flow 24
 - WebSphere DataStage and QualityStage enablement 18
 - job sequence enablement 19
 - sequential file processing 22
 - Single job enablement 18
 - WebSphere Information Analyzer enablement 24
 - Grid Enablement toolkit installation 16
- grid computing in Information Server 6
 - benefits 9
 - grid definitions 9
 - high availability (HA) grid environments 15
 - main components 7
 - typical grid environments 10
 - NAS configuration grid environment 10
 - NAS vs. SAN configurations 13
 - private networks 14
 - SAN configuration grid environments 12
- grid computing overview 2
 - anonymous parallel execution 5
 - commodity hardware and operating systems 5
 - enterprise grid computing 4
 - scientific grid computing 4
 - utility usage model 5
- Grid Enablement toolkit 31, 38, 45, 131, 247
- grid resource manager 7–8

- grid toolkit resource manager commands 254
 - canjob command 254, 258
 - listq command 254
- grid_config.apt 17, 140
- grid_config.apt file 131, 135
- grid_global_values 18, 147
- grid_global_values file 133
- group file 62

H

- head node 9
- high availability 15
- host name 23, 61, 72
- hosts file 61, 129

I

- ifconfig command 60, 72
- IIserver output examples 287
- IIsgGrid.sh script 277
- IISServer.sh script 264
- Information Server
 - grid computing execution flow 16
- install and configure Grid Enablement toolkit
 - execute the test.sh script 136
 - install the toolkit 131
 - restart WebSphere DataStage 133
 - review key files of interest 133
 - source the dsenv file 136
 - toolkit environment variables 139
- install pre-requisite patches
 - patch_e124940 66
 - patch_e126149v4 66
- install.sh script 131, 286
- ISFagents.sh script 270
- ISFserver.sh script 274

J

- job run options 147
- job sequence 19

K

- kickstart file 109, 112–113
- kickstart_template file 113

L

- IGrid toolkit resource manager commands
 - istq command 254

- listq command 204
- listq command 254
- llinit 105
- llmonitor.sh script 260
- llq command 203, 253
- llstatus command 253
- LoadL script 93, 124
- LoadL service 107
- Loadl_admin file 106, 108, 218
- Loadl_config file 105
- Loadl_config.local 128
- Loadl_config.local file 106, 224
- LoadL.sh script 283
- LoadLeveler 102, 107
- logical host name 66
- lspci command 59

M

- MAC ADDR 118
- makefile 82
- managing the Information Server grid environment 37
 - monitoring and tuning 38
 - resource manager 37
 - Troubleshooting 38
- master script 93
- MAX_STARTERS 106, 128
- MaxNodesPerJob 18, 45, 200
- memory 41
- migration scenario
 - business requirement 48
 - environment configuration 49
 - failing back to the original front end node ORION 236
 - failing over to the standby node NILE 221
 - general approach 52
 - installing and configuring the grid environment
 - build a grid environment using the BYOG toolkit 78
 - compile grid environment worksheet 63
 - configure the standby node 143
 - install and configure Grid Enablement toolkit 131
 - install pre-requisite patches 66
 - review existing Information Server environment 55
 - set up Virtual IP and host name and private network 68

- managing the grid environment 200
 - add/remove nodes 217
 - cancel a waiting job from the queue 214
- tailoring existing DS/IA jobs to operate in grid environment 146
 - with parameter & stage property changes 186
 - with parameter changes only 148, 159
- mount filesystems 222
- mounting filesystems 226
- mountsanfs_Nile.sh script 292
- mountsanfs_Orion.sh script 300

N

- nameserver IP address 61
- NAS (network attached storage) 10, 12
- network boot 116, 118
- network file 61, 71
- network.# 110
- network.# file 113
- NFS 16, 78
 - mounts 10, 12, 39, 42, 78
 - script 93, 124
- NIC (network interface card) 12–13, 44, 50, 112, 118
- NIS 80
 - clients 51, 85
 - database 82, 85
 - domain 82
 - domainname 81
 - master server 80, 85
 - script 93, 124
 - slave server 80, 85
- nisupdatefiles.sh script 145, 263
- node 9
- node_table 135
- nodes.cfg file 226
- nsswitch.conf file 63

O

- ONBOOT 69–70
- ONPARENT 145
- osh 17, 24
- osh_conductor.sh script 17, 24
- osh.exe 17

P

- parallel framework 6, 10, 12, 27, 40, 304
 - conductor node 305
 - players 305
 - section leaders 305
- partition 9
- passwd file 62
- physical host name 66, 72
- players 305
- portmap restart command 82
- portmap service 82
- post-env file 93
- primary front end node 143
- primary node 27
- private network 12, 14, 68–70, 112, 135–136
- public network 136
- PXE (Preboot eXecution Environment) 86
 - client 116, 120
- PXE_cfg.sh script 112
- PXEBoot execution 116
- PXEBoot server 86
- pxelinux.cfg 121

R

- range dynamic-bootp 94
- Redbooks Web site 310
 - Contact us xxiv
- remnant processes 227
- remount_Nile.sh script 238
- remount_Nile.sh script 226
- remount_Orion.sh script 238, 301
- remounts_Nile.sh 289, 293
- resource disk entry 12
- resource manager 9, 23, 31, 37
 - queue 141, 203
- resource monitor 9
- reviewing existing Information Server environment
 - /etc/group file contents 62
 - /etc/hosts file contents 61
 - /etc/nsswitch.conf file contents 63
 - /etc/passwd file contents 62
 - directories to be shared 61
 - host name and gateway IP address 61
 - Information Server version and patches 57
 - nameserver IP address 61
 - NIC details and IP address 59
- runcmd.sh script 267

S

- SAN (storage area networks) 10
- sar 38, 142
- scientific grid computing 4
- scripts 78
 - scripts developed 260
 - HA scripts 289
 - failback to ORION scripts 289, 294
 - failback_Nile.sh 289, 294
 - failback_Orion.sh 289, 297
 - mountsanfs_Orion.sh 289, 300
 - remounts_Orion.sh 289, 301
 - umountsanfs_Nile.sh 289, 296
 - failover to NILE scripts 289
 - failover_Nile.sh 289
 - mountsanfs_Nile.sh 292
 - remounts_Nile.sh 289, 293
 - miscellaneous scripts 260
 - IIServer.sh script 260, 264
 - llmonitor script 260
 - nisupdatefiles.sh script 260, 262
 - runcmd.sh script 260, 267
 - services scripts 267
 - dstage.sh script 268
 - IIServer output examples 287
 - IISGrid.sh script 277
 - Install.sh script 286
 - ISFagents.sh script 270
 - ISFserver.sh script 274
 - Loadl.sh script 283
 - uninstall.sh script 288
 - xmeta.sh script 280
 - section leaders 135, 305
 - SELINUX 100, 143
 - sequencer.sh script 19, 26, 186, 188
 - Sequential File stage 19
 - service network status command 59
 - setting up the Information Server grid computing environment 43
 - build and test your grid infrastructure using the BYOG toolkit 45
 - design your grid environment infrastructure 43
 - identify the jobs to be migrated 45
 - install and configure the Grid Enablement toolkit 45
 - set up the management of the grid environment 46
 - tailor existing DS/IA jobs to tolerate or exploit the grid environment 46

- shared work directory 25
- SSH 85
- standby node 15–16, 49, 61, 82, 143, 221–222, 226, 236
- star-schema 48
- StrictHostKeyChecking 86
- successful practices 39

T

- takeover 221
- templates 78
- temporary files 137
- test.sh 136
- test.sh script 18, 131, 137
- TFTP (trivial file transfer protocol) 86, 116
 - service 90, 96
- ttft file 96
- ttftboot directory structure 87

U

- umountsanfs_Nile.sh script 289, 296
- uninstall.sh script 288

V

- virtual IP 55, 68
 - address 143, 145, 221
- vmstat 38

W

- WebSphere DataStage and QualityStage Director 19, 76
- WebSphere Enterprise Framework 85
- WebSphere Information Analyzer job 159
- WebSphereMQConnectorPX stage 175
- Workload Scheduler Load Leveler 102
 - commands 253
 - llq command 253
 - llstatus command 253

X

- xmeta.sh script 280

Y

- ypbind daemon 83
- ypdomainname command 82
- ypinit -m command 83

ypasswd daemon 83
ypserv daemon 83
ypservers file 81
ypxrd daemon 83



Deploying a Grid Solution with the IBM InfoSphere

(0.5" spine)
0.475" <-> 0.875"
250 <-> 459 pages



Deploying a Grid Solution with the IBM InfoSphere Information Server

Deploying the InfoSphere Information Server grid

Migrating an existing InfoSphere Information Server

Using grid deployment toolkits

The IBM InfoSphere Information Server and the IBM Blade/GRID offering is a revolutionary new software/hardware platform (leveraging Linux on Intel) that helps organizations derive more value from the complex heterogeneous information spread across their systems. It enables organizations to integrate disparate data and deliver trusted information wherever and whenever needed, in line and in context, to specific people, applications, and processes.

This IBM Redbooks publication describes a scenario for migrating an existing InfoSphere Information Server parallel framework implementation on a Red Hat Enterprise Linux AS 4 platform to a high availability grid environment involving four machines comprising one conductor node and three compute nodes. The high availability environment involves one of the three compute nodes serving as a standby conductor node that takes over the conductor role when the original conductor fails. The steps involved in migrating the existing infrastructure to the high availability grid infrastructure and enabling existing IBM WebSphere DataStage, IBM WebSphere QualityStage, and IBM WebSphere Information Analyzer jobs to exploit the grid environment are described here.

**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