THE POWER TO KNOW.

Grid Computing in SAS® 9.2
Third Edition

SAS® Documentation

The correct bibliographic citation for this manual is as follows: SAS Institute Inc. 2010. *Grid Computing in SAS® 9.2, Third Edition*. Cary, NC: SAS Institute Inc.

Grid Computing in SAS® 9.2, Third Edition

ISBN 978-1-60764-541-2

SAS Institute Inc., SAS Campus Drive, Cary, North Carolina 27513.

1st electronic book, May 2010

1st printing, May 2010

SAS® Publishing provides a complete selection of books and electronic products to help customers use SAS software to its fullest potential. For more information about our e-books, e-learning products, CDs, and hard-copy books, visit the SAS Publishing Web site at **support.sas.com/publishing** or call 1-800-727-3228.

Contents

What's New in SAS Grid Manager 9.2

Overview

SAS Grid Manager has the following new features and enhancements:

- A SAS code analyzer is added to automatically add syntax to existing SAS programs in order to enable parallel processing on a grid.
- High-availability capabilities are provided as part of SAS Grid Manager.
- A method for submitting batch SAS jobs to the grid has been added.
- The capability for SAS Grid Manager to provide load balancing for SAS workspace servers has been added.
- Job control has been enhanced.
- Enhancements to the Grid Manager plug-in for SAS Management Console provide improved grid monitoring and control.
- Enhancements to the Schedule Manager plug-in for SAS Management Console provide improved control and monitoring for jobs and flows scheduled to run on the grid.
- Support is added for other grid middleware providers.

SAS Code Analyzer

The SAS Code Analyzer is a procedure that executes an existing SAS program and identifies the dependencies of the procedures and job steps. SAS Code analyzer then uses this information to create a new version of the program that contains the syntax required for the subtasks to be executed in parallel on a grid.

High-Availability Capabilities

High-availability capabilities are incorporated into Platform Suite for SAS 4.1. This capability provides high availability for critical components running in a grid (such as the

SAS Metadata Server) and eliminates the need for a hot standby machine and the purchase of additional third-party tools.

A New Way to Submit Batch SAS Jobs to the Grid

The second maintenance release after SAS 9.2 adds the SAS Grid Manager Client Utility. This utility enables batch SAS jobs to be submitted to the grid without the need to have SAS installed on the machine that is used to submit the jobs. The utility provides the capability to submit jobs, end jobs, check job status, and retrieve job output.

Grid Algorithm for Load Balancing

SAS Grid Manager can be used to provide load balancing for workspace servers running in a grid. This capability provides a robust way to enable load balancing for any clients that use SAS workspace servers.

Enhanced Job Control

The following enhancements improve control for jobs processed on a SAS grid:

- A job name can be specified through a macro variable specified by the JOBNAME option of the GRDSVC_ENABLE statement.
- Job options can be specified through a macro variable specified by the JOBOPTS option of the GRDSVC_ENABLE statement.
- Job options can be specified in metadata for grid logical server definitions. These options override user-specified options.
- SAS startup options can be specified in metadata for grid logical server definitions.

Enhancements to the Grid Manager Plug-In

The following enhancements to the Grid Manager plug-in for SAS Management Console provide improved grid monitoring and control:

- The plug-in provides Gantt charts to display job information by status or host.
- Capabilities are provided to:
  - suspend and resume jobs
  - open and close hosts
  - open, close, activate, and deactivate queues

Enhancements to the Schedule Manager Plug-In

The following enhancements to the Schedule Manager plug-in for SAS Management Console provide improved control for jobs and flows that are scheduled to run on a SAS grid:

- Enhancements to the table view provide more information about scheduled jobs, and the ability to filter the contents and view the SAS log.
- A new visual editor improves the process of creating and editing flows to be scheduled.
- Enhancements to the management of deployed flows provide the ability to create and edit trigger events and execution attributes.
- The ability to redeploy a job for scheduling has been added.
- Management of deployed jobs has been enhanced, including the ability to change the batch server and specifying the associated job.

Support for Other Grid Middleware Providers

Support is added for DataSynapse GridServer and Univa UD Grid MP as grid middleware providers. Platform Suite for SAS remains the middleware provider that is packaged with SAS Grid Manager. DataSynapse and Univa UD support includes multi-user load balancing and parallel load balancing. It does not include an interface to the schedule manager framework.

Recommended Reading

- *SAS/CONNECT User's Guide*
- *SAS Deployment Wizard User's Guide*
- *SAS Intelligence Platform: Installation and Configuration Guide*
- *SAS Language Reference: Dictionary*
- *SAS Macro Language: Reference*
- *Scheduling in SAS*

For a complete list of SAS publications, go to support.sas.com/bookstore. If you have questions about which titles you need, please contact a SAS Publishing Sales Representative at:

SAS Publishing Sales
SAS Campus Drive
Cary, NC 27513
Telephone: 1-800-727-3228
Fax: 1-919-531-9439
E-mail: sasbook@sas.com
Web address: support.sas.com/bookstore

Part 1

Grid Computing for SAS

Chapter 1

What Is SAS Grid Computing?

SAS Grid Computing Basics

A SAS grid computing environment is one in which SAS computing tasks are distributed among multiple computers on a network, all under the control of SAS Grid Manager. In this environment, workloads are distributed across a grid of computers. This workload distribution enables the following functionality:

Workload balancing
: enabling multiple users in a SAS environment to distribute workloads to a shared pool of resources.

Accelerated processing
: allowing users to distribute subtasks of individual SAS jobs to a shared pool of resources. The grid enables the subtasks to run in parallel on different parts of the grid, which completes the job much faster.

Scheduling jobs
: allowing users to schedule jobs, which are automatically routed to the shared resource pool at an appropriate time.

SAS Grid Manager provides load balancing, policy enforcement, efficient resource allocation, and prioritization for SAS products and solutions running in a shared grid environment. It also separates the SAS applications from the infrastructure used to execute the applications. This enables you to transparently add or remove hardware resources as needed and also provides tolerance of hardware failures within the grid infrastructure. SAS Grid Manager integrates the resource management and scheduling capabilities of the

Platform Suite for SAS with the SAS 4GL syntax and subsequently with several SAS products and solutions.

SAS Grid Manager includes these components, as illustrated in Figure 1.1 on page 5. :

Grid Manager plug-in
: a plug-in for SAS Management Console that provides a monitoring and management interface for the jobs and resources in your grid

grid syntax
: the SAS syntax necessary to grid-enable the SAS workload

Platform Suite for SAS
: components provided by Platform Computing to provide efficient resource allocation, policy management, and load balancing of SAS workload requests

The Platform Suite for SAS includes these components:

Load Sharing Facility (LSF)
: dispatches all jobs submitted to it, either by Process Manager or directly by SAS, and returns the status of each job. LSF also manages any resource requirements and performs load balancing across machines in a grid environment.

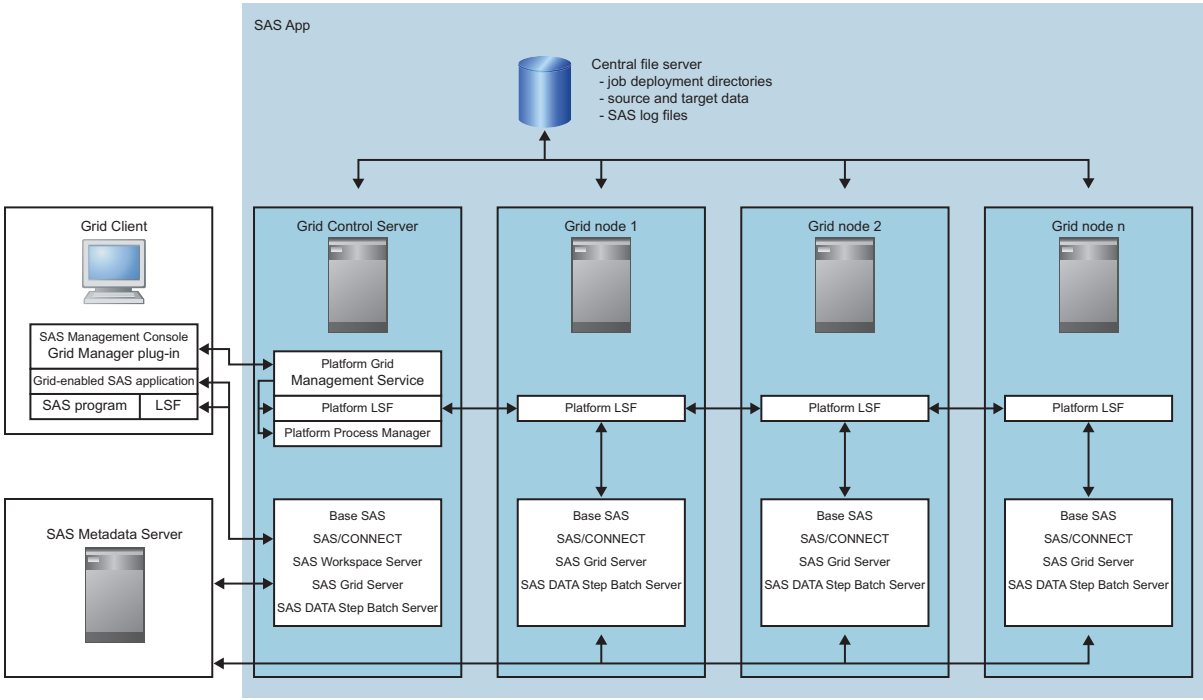
Process Manager (PM)
: this is the interface used by the SAS scheduling framework to control the submission of scheduled jobs to LSF and manage any dependencies between the jobs.

Grid Management Services (GMS)
: this is the interface to the Grid Manager plug-in in SAS Management Console. It provides the run-time information about jobs, hosts and queues for display in SAS Management Console.

SAS Grid Topology

As illustrated below, a grid configuration consists of these main components:

Figure 1.1 *Grid Topology*



Grid control server
: this machine controls distribution of jobs to the grid. Any machine in the grid can be designated as the grid control server. Also, you can choose whether to configure the grid control server as a grid resource capable of receiving work. This machine must contain the grid middleware software (such as Platform Suite for SAS). The grid control server also configures a SAS workspace server so that SAS Data Integration Studio and SAS Enterprise Miner can run programs that take advantage of the grid.

Grid node
: these machines are grid computing resources that are capable of receiving the work that is being distributed to the grid. The number of nodes in a grid depends on the size, complexity, and volume of the jobs that will be run by the grid You can add or remove nodes as specified by your business needs. Each grid node must contain Base SAS, SAS/CONNECT, Platform LSF (or other grid middleware software), and any applications and solutions needed to run grid-enabled jobs.

Central file server
: this machine is used to store data for jobs that run on the grid. In order to simplify installation and ease maintenance, you can also install the SAS binaries on the central file server.

Metadata server
: this machine contains the metadata repository that stores the metadata definitions needed by SAS Grid Manager and other SAS applications and solutions that are running on the grid. Although it is recommended that the SAS Metadata Server be on a dedicated machine, it can be run on the grid control server.

SAS Management Console
: this application is used to manage the definitions in the metadata repository, to submit jobs to the grid through the Schedule Manager plug-in, and to monitor and manage the grid through the Grid Manager plug-in.

Grid clients
: submits jobs to the grid for processing, but is not part of the grid resources available to execute work.

  Examples of grid clients are:

  - a SAS Data Integration Studio client. Platform LSF is not required on this client machine.
  - a SAS Enterprise Miner client. Platform LSF is not required on this client machine.
  - a SAS Management Console client, that uses the Schedule Manager plug-in or another application to schedule SAS workflows. Platform LSF is not required on this client machine.
  - a SAS foundation install that is used to run a program that submits work to the grid. The submitted work can be entire programs or programs broken into parallel chunks. This client must have Base SAS, SAS/CONNECT, and Platform LSF installed. Platform LSF is required to submit the SAS workload to the grid.
  - a SAS Grid Manager Client Utility. SAS is not required to be installed on this client, but Platform LSF is required to submit the SAS workload to the grid.

What Types of Processing Does a Grid Support?

Multi-User Workload Balancing

Most organizations have many SAS users performing a variety of query, reporting, and modeling tasks and competing for the same resources. SAS Grid Manager can help bring order to this environment by providing capabilities such as the following:

- specifying which jobs get priority
- deciding the share of computing resources used by each job
- controlling the number of jobs that are executing at any one time

In practice, SAS Grid Manager submits work to the grid middleware, which acts as a gatekeeper for the jobs submitted to servers. As jobs are submitted, the middleware (such as Platform LSF) doles them out to grid nodes, preventing any one machine from being overloaded. If more jobs are submitted than can be run at once, the grid middleware submits as many jobs as can be run. The middleware then holds the rest in a queue until resources are free. The grid middleware can also use job priority to determine whether a job is run immediately or held in a queue.

The application user notices little or no difference when working with a grid. For example, users can define a key sequence to submit a job to a grid rather than running it on their local workstation. Batch jobs can be run using wrapper code that adds the commands needed to run the job in the grid. SAS Enterprise Guide applications can be set up to automatically insert the code needed to submit the job to the grid.

Parallel Workload Balancing

Some SAS programs consist of subtasks that are independent units of work and can be distributed across a grid and executed in parallel. You can use SAS syntax to identify the parallel units of work in these programs, and then use SAS Grid Manager to distribute the

programs across the grid. Using parallel workload balancing can substantially accelerate the entire application.

Applications such as SAS Data Integration Studio, SAS Risk Dimensions, and SAS Enterprise Miner are often used for iterative processing. In this type of processing, the same analysis is applied to different subsets of data or different analysis is applied to a single subset of data. Using SAS Grid Manager can improve the efficiency of these processes, because the iterations can be assigned to different grid nodes. Because the jobs run in parallel, the analysis completes more quickly and with less strain on computing resources.

Distributed Enterprise Scheduling

The Schedule Manager plug-in for SAS Management Console provides the ability to schedule user-written SAS programs as well as jobs from numerous SAS applications. You can schedule the jobs and programs to run when specified time or file events occur. If jobs are scheduled using the scheduling capabilities provided by Platform Suite for SAS, the jobs can be processed on a grid without any change to the scheduling process. This capability provides further control over use of computing resources, because you can use the scheduling capability to control when a job runs and the SAS Grid Manager capability to determine which computing resource processes the job.

SAS Applications That Support Grid Processing

The following table lists the SAS applications that currently support grid processing and the type of processing that each supports.

Table 1.1 *Grid Support in SAS Applications*

SAS Application	Multi-User Workload Balancing	Parallel Workload Balancing	Distributed Enterprise Scheduling
Any SAS program	yes	yes, with modifications	yes
SAS Enterprise Guide	yes, with modifications		
SAS Data Integration Studio	yes	yes	yes
SAS Enterprise Miner	yes	yes	
SAS Risk Dimensions	yes	yes, with modifications	
SAS Web Report Studio			yes
SAS Marketing Automation			yes
SAS Marketing Optimization			yes

SAS Application	Multi-User Workload Balancing	Parallel Workload Balancing	Distributed Enterprise Scheduling
SAS JMP/Genomics		yes	
SAS Demand Forecasting for Retail		yes	
SAS products or solutions that use workspace server load balancing	yes		
SAS stored processes	yes, with limitations	yes, with limitations	

For a current list of SAS applications that support grid processing, see **http://support.sas.com/rnd/scalability/grid/index.html**.

What Business Problems Can a Grid Solve?

Many Users on Single Resource

An organization might have multiple users submitting jobs to run on one server. When the environment is first configured, the server might have been sufficient to handle the number of users and jobs. However, as the number of users submitting jobs grows, the load on the server grows. The increased load might lead to slower processing times and system crashes. In a SAS grid environment, jobs are automatically routed to any one of the servers on the grid. This spreads the computing load over multiple servers, and diminishes the chances of a server becoming overloaded. If the number of jobs exceeds the resources available, the jobs are queued until resources become available. If the number of users continues to increase, you can increase capacity by adding servers to the grid.

Increased Data Growth

Your organization might have a process running to analyze a certain volume of data. Although the server that is processing the job is sufficient to handle the current volume of data, the situation might change if the volume of data increases. As the amount of data increases, the load on the server increases, which can lead to longer processing times or other problems. Changing to a larger-capacity server can involve considerable expense and service interruption.

A SAS grid environment can grow to meet increases in the amount of data processed. If the volume of data exceeds the capacity of a server on the grid, the processing load can be shared by other grid servers. If the volume continues to increase, you can add servers to the grid without having to make configuration changes to your processes. Adding servers to the grid is also more cost-effective than replacing a single large server, because you can add smaller servers to handle incremental increases in data volume.

Running Larger and More Complex Analysis

Your organization might have a process running to perform a certain level of analysis on data. If you want to increase the complexity of the analysis being performed, the increased workload puts a greater strain on the processing server. Changing the computing power of the server involves considerable expense and interrupts network availability.

Using a SAS grid environment enables you to add computing power by adding additional computers on the grid. The analysis job can be divided up among the grid nodes, which enables you to perform more complex analysis without increasing the load on any single machine.

Need for a Flexible IT Infrastructure

Your organization's ability to perform the data analysis you need depends on a flexible computing infrastructure. You must be able to add needed resources quickly and in a cost-effective manner as the load increases. You must also be able to handle maintenance issues (such as adding or replacing resources) without disrupting your work. A SAS grid environment enables you to maintain a flexible infrastructure without disrupting your operations.

As your data-processing needs grow, you can incrementally add computing resources to your grid by adding smaller, less-expensive servers as new server nodes. This ability prevents you from having to make large additions to your environment by adding large and expensive servers.

When you need to perform maintenance on machines in the grid, the grid can still operate without disruption. When you take the servers offline for maintenance or upgrades, SAS Grid Manager routes to work to the machines that are still online. Users who send work to the grid for processing do not have to change their way of working. Work that is sent to the grid is processed just as before.

Likewise, the SAS grid environment adapts if a computer fails on the grid. Because SAS Grid Manager automatically avoids sending work to the failed machine, the rest of the grid is still available for processing and users do not see any disruption.

Chapter 2

Planning and Configuring a Grid Environment

Installation and Configuration Overview

The process of configuring a grid consists of two main tasks:

1. Installing and configuring the grid middleware such as Platform Suite for SAS. Instructions for installing and configuring Platform Suite for SAS are found on the SAS Web site at **http://support.sas.com/rnd/scalability/grid/gridinstall.html**
2. Installing and configuring SAS products and metadata definitions on the grid. You can either install all SAS products on all machines in the grid or install different sets of SAS applications on sets of machines in the grid. However, Base SAS and SAS/CONNECT must be installed on all grid machines. Using a grid plan file with the SAS Deployment Wizard guides you through the process of installing and configuring the SAS applications and metadata definitions on each machine in the grid. It is recommended that you specify the same directory structure on all machines in the grid.

For information about performing a planned installation, see *SAS Intelligence Platform: Installation and Configuration Guide*.

Configuring the File Server

The central file server is a critical component of a grid environment. It is essential for each application on a grid node to be able to efficiently access data. Slowdowns caused by the performance of the file storage system could reduce the effectiveness and benefit of using a grid. The amount of storage required and the type of I/O transactions help to determine the type of file storage system that best meets your needs.

Assuming that the SAS jobs running on the grid perform an equal number of reads and writes, it is recommended that the file system be able to sustain 25–30 MB/seconds per core. This level can be adjusted up or down, depending on the level of I/O activity of your SAS jobs. For information about choosing and configuring a file system, see *Best Practices for Data Sharing in a Grid Distributed SAS Environment*, which is available at **http://support.sas.com/rnd/scalability/grid/gridpapers.html.**

Installing the Grid Middleware

SAS Grid Manager includes Platform Suite for SAS from Platform Computing. SAS Grid Manager also supports Univa UD Grid MP or DataSynapse GridServer to provide multi-user workload balancing and parallel workload balancing.

The SAS Web site provides step-by-step instructions on installing and configuring the Platform Suite for SAS and information about configuring other grid middleware. These instructions are available from **http://support.sas.com/rnd/scalability/grid/gridinstall.html.**

Information for installing Platform Suite for SAS is available for both Windows and UNIX platforms.

The installation process for Platform Suite for SAS installs these components:

- Platform Process Manager
- Platform LSF
- Platform Grid Management Service

Configuring the Grid Control Server

After you install and configure the grid middleware, you can use the SAS Deployment Wizard to configure the grid control server. The SAS Deployment Wizard installs and configures these components:

Table 2.1 *SAS Deployment Wizard Grid Control Server Components*

Installed SAS Software Components	Configured SAS Software Components
• SAS Foundation (including Base SAS and SAS/CONNECT) • SAS Management Console • Grid Manager Plug-in for SAS Management Console	• Platform Process Manager Server • Grid Monitoring Server • SAS Application Server (SAS Logical DATA Step Batch Server, SAS Logical Grid Server, SAS Logical Workspace Server) • Object Spawner • Grid script file

If you are installing Platform Suite for SAS on a UNIX machine, you might need to source the profile.lsf file before you start the SAS Deployment Wizard. The hostsetup command in the installation procedure for Platform LSF version 7 includes the ability to source the LSF profile to the default profile for all users. If this option was not used in the installation process or did not work correctly, you must use the following procedure. This procedure enables the SAS Deployment Wizard to find the addresource utility. To source the file, follow these steps:

1. Start the LSF daemons. The easiest method for doing this is to reboot the computer on which Platform Suite for SAS is installed.
2. Using the default profile for the machine, issue this command:

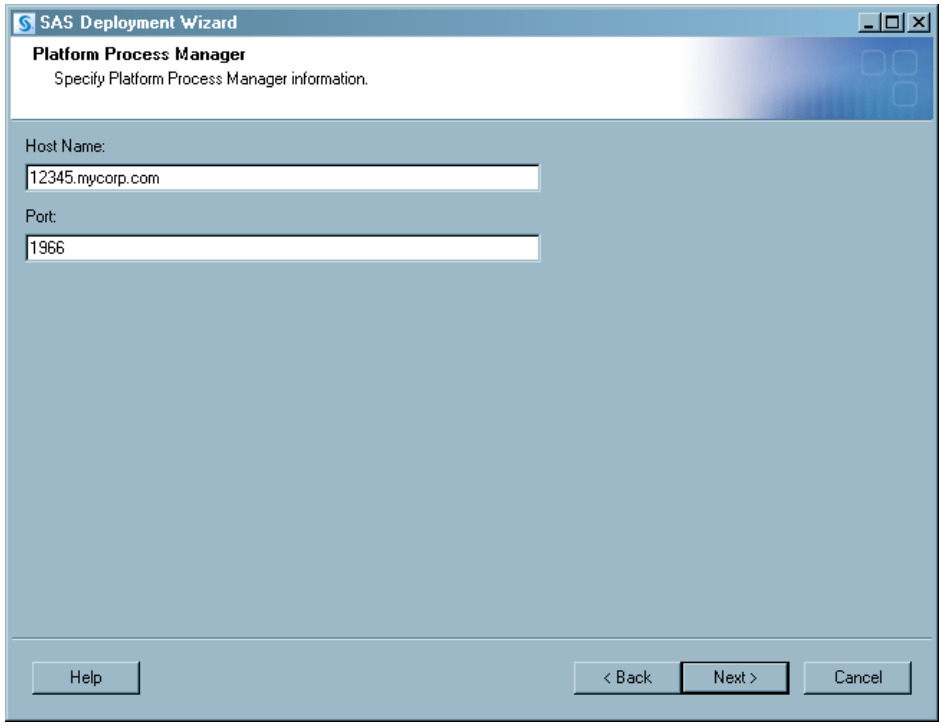
   ```
   . LSF_TOP/conf/profile.lsf
   ```

   Replace LSF_TOP with the directory in which Platform LSF is installed. Note that the command starts with a period.

The amount of user input that is required during the installation and configuration process depends on whether you choose an Express, Typical, or Custom install. For information about running the SAS Deployment Wizard, see *SAS Deployment Wizard User's Guide*.

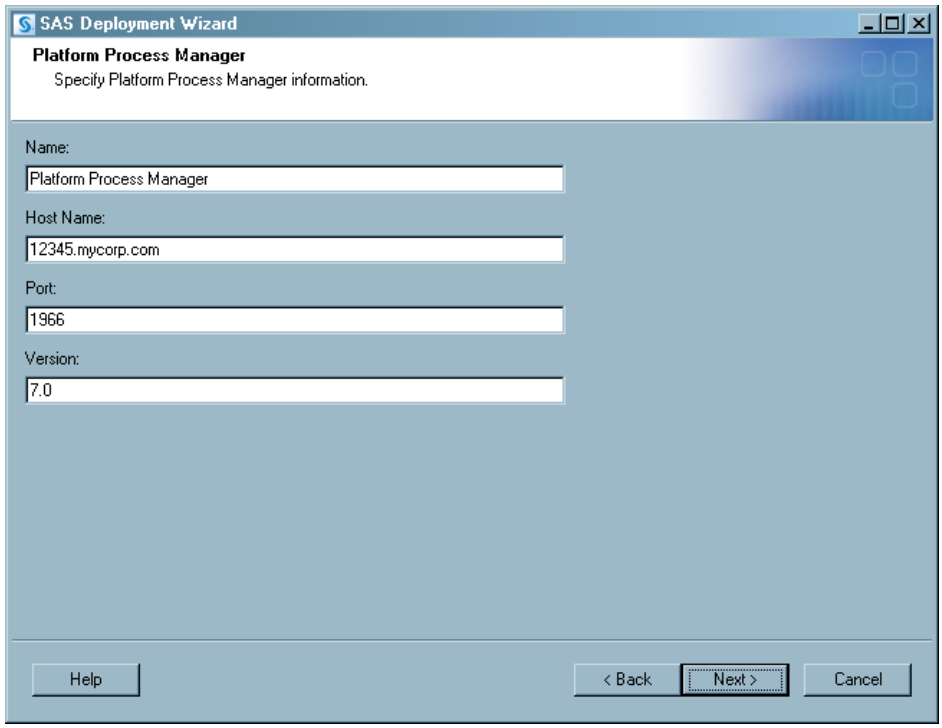
An Express installation does not request any grid-specific information. Default values are used in all cases, so you must verify that these values match the values needed for your environment

The Platform Process Manager information page enables you to specify the host name and port of the machine on which Platform Process Manager installed.

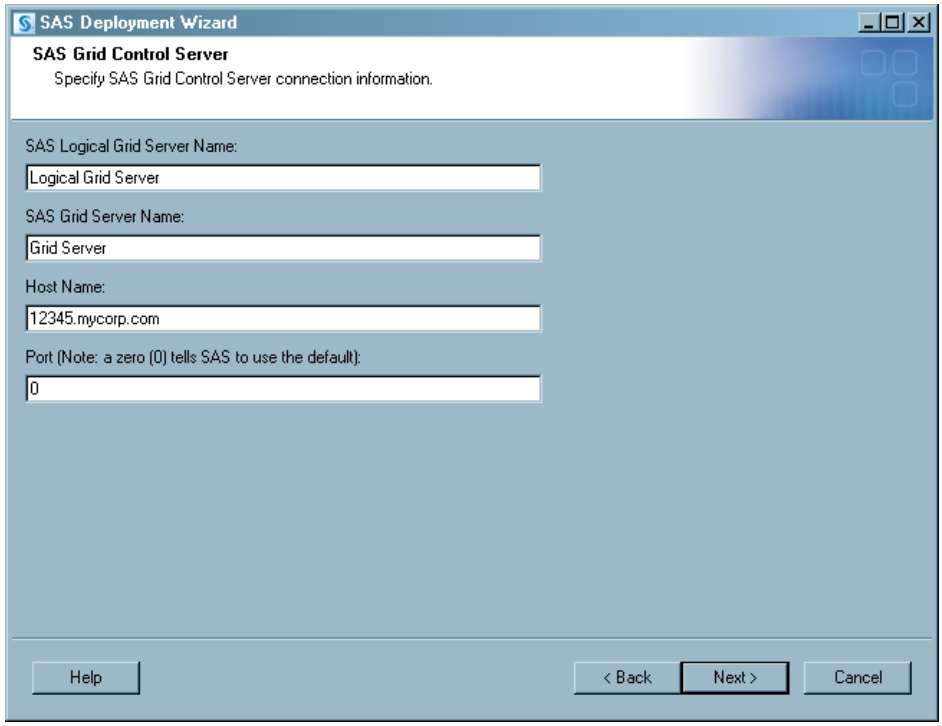


During the installation and configuration process for a Custom install, the SAS Deployment Wizard displays these pages that request grid-specific information:

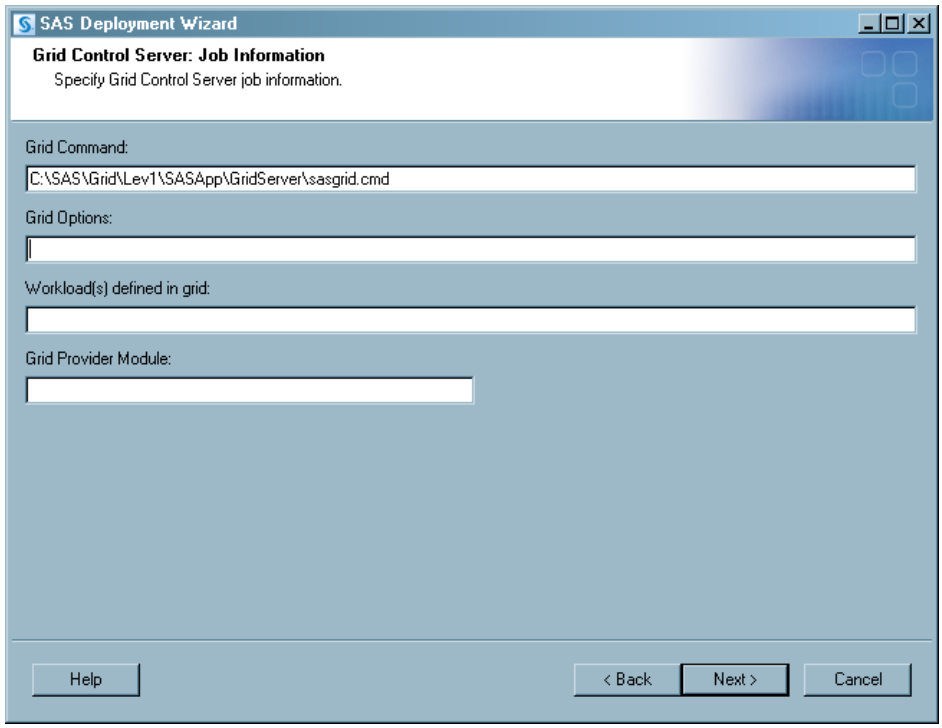
1. The Platform Process Manager information page enables you to specify the server on which you installed Platform Suite for SAS and the port used to connect to the server.



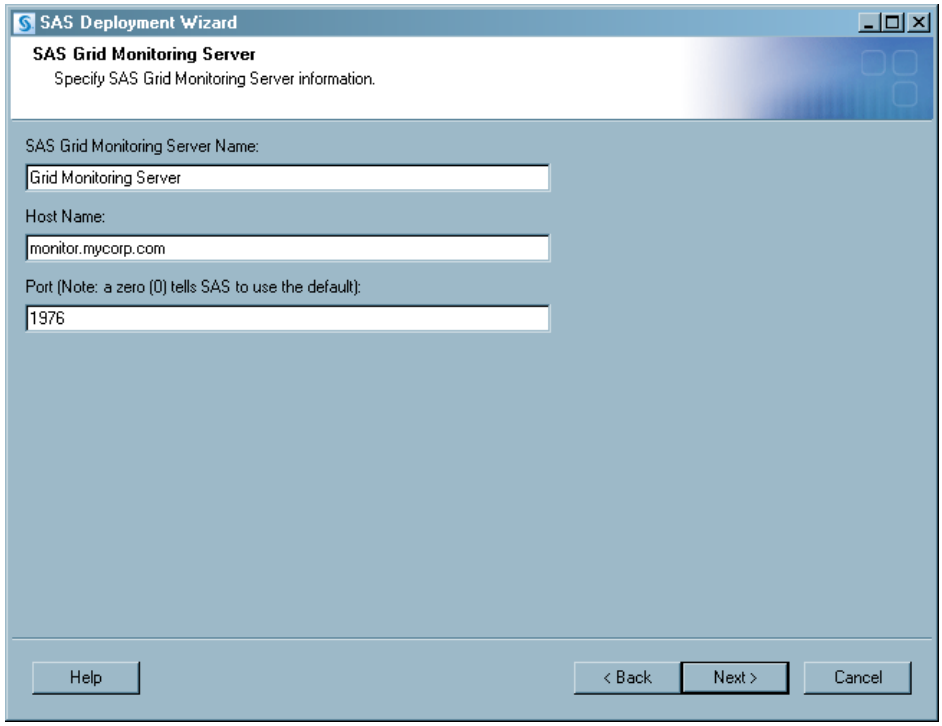
2. The SAS Grid Control Server information page enables you to specify the name of the SAS Logical Grid Server and the SAS Grid Server. Specify the grid control server machine and port number. For Platform Suite for SAS, specify a value of `0` in the **Port** field.



3. The Grid Control Server Job Information page enables you to specify how jobs run on the grid. Specify the command used to start the server session on the grid, workload values, and additional options for the grid. For information about the values used in these fields, see "Modifying SAS Logical Grid Server Definitions" on page 17.



4. The SAS Grid Monitoring Server page enables you to specify the name, machine, and port for the grid monitoring server.



Configuring the Grid Nodes

After you have installed and configured the grid control server, you can use the SAS Deployment Wizard to configure the grid nodes. The SAS Deployment Wizard installs and configures these components:

Table 2.2 *Required Software Components for Grid Nodes*

Installed SAS Software Components	Configured SAS Software Components
SAS Foundation (Base SAS, SAS/CONNECT)	SAS Grid Node, script file

The amount of user input that is required during the installation and configuration process depends on whether you choose an Express, Typical, or Custom install. For information about running the SAS Deployment Wizard, see *SAS Deployment Wizard User's Guide*.

For information about the values required during a planned installation, see *SAS Intelligence Platform: Installation and Configuration Guide*.

Note: The configuration directory structure for each grid node must be the same as that of the grid control server.

Configuring Client Applications

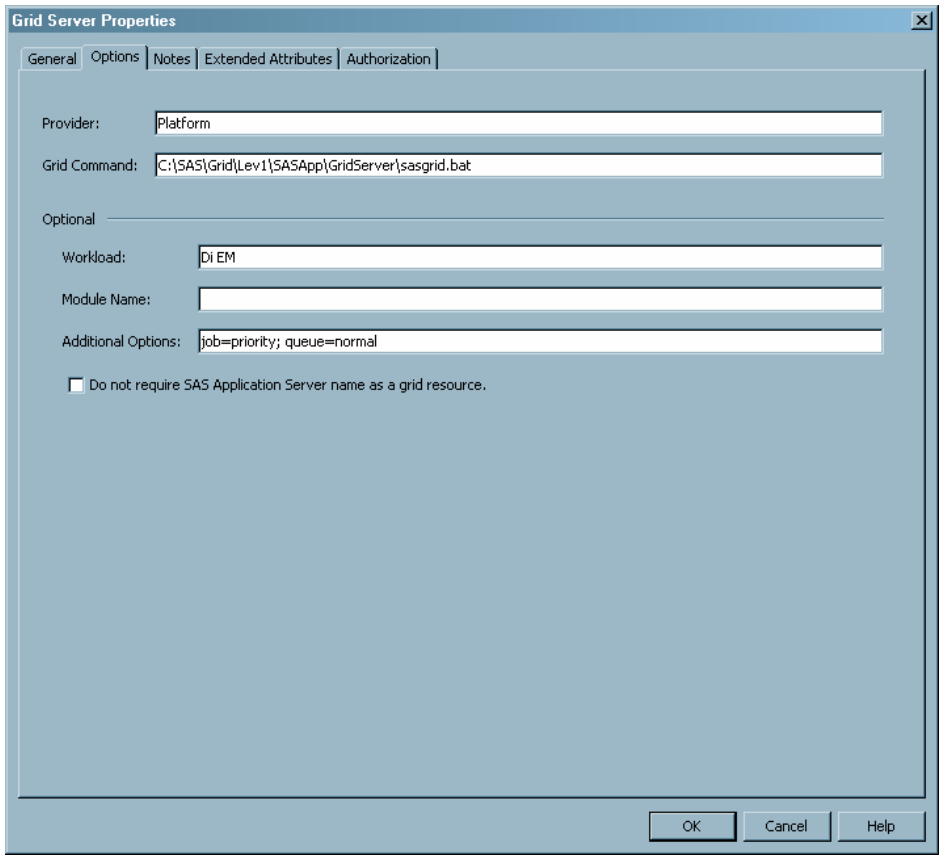
After the grid nodes have been installed and configured, you can install and configure the software required for the client applications that will use the grid. The software required will depend on the type of client application. Applications such as SAS Data Integration Studio that can submit jobs through a workspace server do not need to install anything other than the client application. Applications such as Base SAS that submit jobs to the grid must also install Platform Suite for SAS or other middleware in order to send jobs to the grid. When you install SAS Management Console, which is used to monitor and control the grid, you must also install the SAS Grid Manager plug-in.

Modifying SAS Logical Grid Server Definitions

The initial configuration of the logical grid servers are performed by the SAS Deployment Wizard. However, a SAS grid administrator might need to modify the existing grid metadata or add new grid metadata definitions.

A SAS administrator performs these steps to specify or modify the required and optional properties as metadata for the SAS Grid Server:

1. In SAS Management Console, open the metadata repository that contains the metadata for the Logical Grid Server.
2. In the navigation tree, select **Server Manager**.
3. Expand the folders under **Server Manager** until you see the metadata objects for the SAS application server, such as **`SASApp`**, and its Logical Grid Server component.
4. Expand the Logical Grid Server component so that you see the metadata object for the Grid Server.
5. Right-click the metadata object for the Grid Server, and select **Properties**.
6. In the Properties window for the Grid Server, click the **Options** tab.



7. The values for each field are different according to the grid middleware provider you use. This section lists the values used with Platform Suite for SAS. See **http://support.sas.com/rnd/scalability/grid/gridinstall.html** for values for other middleware providers. The fields on the Options tab are:

Provider
the grid middleware provider. For Platform Suite for SAS, this value is Platform. This value is used to communicate with the grid control server.

Grid Command
the script, application, or service that the grid middleware uses to start server sessions on the grid nodes.

For the Platform Suite for SAS, this value is the path to the sasgrid.cmd file (Windows) or sasgrid script file (UNIX). Because this same command is used to start the servers on all grid nodes, the path to the directory on each grid node must be the same. For example:
C:\SAS\Grid\Lev1\SASApp\GridServer\sasgrid

Workload
a user-defined string that specifies the resources or the types of jobs that can be processed on the grid. For example, the grid administrator could create resources named di_short and di_long for short- and long-running SAS Data Integration Studio jobs. By placing those values in this field, SAS Data Integration Studio users can select one of those values from the SAS Data Integration Studio options dialog boxes.See "Using SAS Data Integration Studio with a SAS Grid " on page 44. After the values are selected, the value is sent with the job to the grid so that the job runs only on the machines that have the specified resource defined.

Workload values can be separated by a space. For information about specifying resources, see "Partitioning the Grid " on page 28.

Module Name
specifies the shared library name or the class name of the middleware provider's support plug-in. Leave blank unless directed otherwise by SAS Technical Support.

Additional Options
the options used by the SAS command to start a session on the grid node or to control the operation of the job. For Platform Suite for SAS, examples include the job priority, the job queue, or user group that is associated with the job. Job options are specified as name/value pairs in this format:

```
option-1=value-1;option-2="value-2 with spaces";
```

```
... option-n='value-n with spaces';
```
Here is an example of additional options for Platform Suite for SAS. These options specify that all jobs that use this logical grid server go to the priority queue in the project "payroll":

```
queue=priority; project='payroll'
```

For a complete list of job options, see "Supported Job Options " on page 89.

Do not require SAS Application Server name as a grid resource
if selected, specifies that the SAS Application Server name is not used by the grid to determine which grid node processes the requests. If this check box is cleared, the SAS Application Server name is included as a required resource. This option is typically not selected. Select this option if you are implementing a SAS floating license grid and no resources are defined on the individual grid nodes. For more information, see "Removing the Resource Name Requirement" on page 30.

8. After you complete the field entries, click **OK** to save the changes and close the Grid Server Properties window.
9. In the display area (right-hand side) on SAS Management Console, right-click the Connection object for the Grid Server, and then select **Properties**.
10. In the Properties window for the Grid Server Connection, click the **Options** tab. The fields on this tab are:

Authentication Domain
the authentication domain used for connections to the server.

Grid Server Address
the host name or network address of the grid control server.

Grid Server Port
the port used to connect to the grid control server. If this is set to 0 (zero), the default port for the grid provider is used (if a default value exists).

Modifying Grid Monitoring Server Definitions

The initial configuration of the grid monitoring server is performed by the SAS Deployment Wizard. However, a SAS grid administrator might need to modify the existing grid metadata or add new grid metadata definitions.

A SAS administrator performs these steps to specify or modify the required and optional properties as metadata for the Grid Monitoring Server:

1. In SAS Management Console, open the metadata repository that contains the metadata for the SAS Grid Server.
2. In the navigation tree, select **Server Manager**.

3. Find the metadata object for the Grid Monitoring Server.
4. Right-click the metadata object for the Grid Monitoring Server, and then select **Properties**.
5. In the Properties window for the Grid Monitoring Server, click the **Options** tab.
6. The values for each field are different according to the grid middleware provider you use. This section generally lists the values used with Platform Suite for SAS. See `http://support.sas.com/rnd/scalability/grid` for values for other middleware providers. The fields on the Options tab are:

   Provider
   the grid middleware provider. For Platform Suite for SAS, this value is Platform. This value is used to communicate with the grid control server.

   Module Name
   specifies the shared library name or the class name of the middleware provider's support plug-in. Leave this field blank unless directed otherwise by SAS Technical Support.

   Options
   the options needed by the grid monitoring server to connect to the grid server.
7. After you complete the field entries, click **OK** to save the changes and close the Grid Monitoring Server Properties window.
8. In the display area (right side) on SAS Management Console, right-click the Connection object for the Grid Monitoring Server, and then select **Properties**.
9. In the Properties window for the Grid Monitoring Server Connection, click the **Options** tab. The fields on this tab are:

   Authentication Domain
   the authentication domain used for connections to the server.

   Host Name
   the network address of the grid control server.

   Port
   the port used to connect to the grid control server. If set to 0 (zero), the default port for the grid provider is used (if a default value exists).
10. After you complete the entries, click **OK** to save the changes and close the Grid Monitoring Server Connection Properties window.

Naming the WORK Library

If you are using a shared file system for the SASWORK libraries created by each SAS grid session, each SASWORK subdirectory must have a unique name. The default method used by SAS to generate unique work directories does not maintain unique directories across grid nodes.

To ensure unique work directory names across grid nodes, you can add a machine name component to the `-work` parameter in the Grid Command field of the Grid Server metadata definition. Alternatively, you could include the parameters in the sasgrid.cmd file (on Windows) or the sasgrid file (on UNIX).

An example command is `-work S:\SASWork\%COMPUTERNAME%`.

An example invocation line is: **"C:\Program Files\SAS\SASFoundation \9.2\sas.exe" %SASCFGPATH% %SASCFGLOGFILE% -dmr -nologo -noterminal -nosyntaxcheck -icon -work . -sasuser -ipaddress -metaautoresources "SASApp" %SASUSERARGS% -work S:\SASWork\ %COMPUTERNAME%**

Installing and Configuring SAS Grid Manager Client Utility

Installation Overview

The SAS Grid Manager Client Utility has been added in the second maintenance release after SAS 9.2. This utility enables users to submit SAS programs to a grid for processing without having SAS installed on the machine performing the submission.

If you install SAS Grid Manager for the first time using the second maintenance release after SAS 9.2, the SAS Grid Manager Client Utility is automatically installed and configured using the SAS Deployment Wizard if the utility is in the plan file.

Installation Prerequisites

The configuration for the SAS Grid Manager Client Utility assumes that all of the following actions have been performed:

- The grid control server has already been installed. The configuration must retrieve the logical grid server definition from metadata.
- The user name under which jobs are submitted is defined in metadata. If not, jobs submitted to the grid will fail.
- A shared directory or shared file system is available to the client machine and the grid machines.
- You have copied the SID file used to install the grid control server to the GRIDWORK directory and you have renamed the file to license.sasgsub. The SID file must have the Grid Manager product enabled in it.

Using the SASGSUB Configuration File

Most of the options that are used by the SAS Grid Manager Client Utility are contained in the sasgsub.cfg file, which is automatically created by the SAS Deployment Wizard. These options specify the information that the SAS Grid Manager Client Utility uses every time it runs. The sasgsub.cfg file is located in the Applications/SASGridManagerClientUtility/<*version*> directory of the configuration directory. The following information from the SAS Deployment Wizard is collected in the sasgsub.cfg file:

- information to connect to the SAS Metadata Server (SAS Metadata Server name, port, user ID, and password). By default, the metadata password value is set to _PROMPT_, and the user is prompted for a password.
- the path to the shared file system used to share files between the user and the grid.
- the name of the SAS Application Server that contains the logical grid server definition.

If you are using a grid provider other than Platform Suite for SAS and are using the SAS Deployment Wizard in Expert mode, you can also specify these options:

- the grid user and password, if required by the grid provider that you are using. If you specify the user name, the default value for the password is _PROMPT_, and the user is prompted for a password.
- the path to any additional JAR files required by the grid provider.

The SAS Grid Manager Client Utility configuration assumes a location and name for the license file containing the SAS Grid Manager license. Move the SID text file to the GRIDWORK directory and rename the file to license.sasgsub.

Modifying the SASGRID Script File

If you installed your grid using SAS 9.2 before the second maintenance release, you must change the sasgrid script file. Follow these steps to change the file:

1. Edit the sasgrid.cmd file (Windows) or the sasgrid file (UNIX). The file is located in the GridServer directory under the configuration directory.

   Note: If you are using Windows, the editor that you use must save the file using carriage return/line feed as the line termination characters.

2. Locate the SASEXEFILE environment variable and change the @sas.exec.file@ value to the path to the SAS executable file on all of the grid machines. For example, you might change the Windows sasgrid.cmd file from

   ```
   set SASEXEFILE="@sas.exec.file@"
   ```

   to

   ```
   set SASEXEFILE="C:\Program Files\SAS\SASFoundation\9.2\sas.exe"
   ```

3. Save the file.

4. Copy the script file to each machine in the grid. The file should be located in the GridServer directory under the SAS configuration directory associated with the SAS Application Server used by the SAS Grid Manager Client Utility. For example, under Windows, you should copy the sasgrid.cmd file to **`C:\SASConfig\Grid\Lev1\SASApp\GridServer`** if the SAS configuration directory is **`C:\SASConfig\Grid\Lev1`** and the application server is SASApp.

   Note: If you do not update the script file on all machines in the grid, the SASGSUB -GRIDGETSTATUS command does not report the correct status for a job submitted to the grid. The job always appears to be in a “Submitted” state.

Chapter 3

Managing the Grid

Overview of Grid Management

Most organizations that use SAS consist of a variety of categories of users, with each category having its own needs and expectations. For example, your organization might have these users:

SAS Enterprise Guide users
: these users are usually running interactive programs, and expect immediate results.

SAS Enterprise Miner users
: these users might be using multiple machines to train models.

SAS Web Report Studio users
: these users might be scheduling reports to run at a specified time.

SAS Risk Dimensions users
: these users might be running jobs at night.

Some users in your environment might be running jobs that have a high priority. Other users might be running jobs that require a large number of computing resources. A SAS grid environment must be able to account for all of these different needs, priorities, and workloads.

In order to manage this type of environment, you must be able to control when and where jobs can run in the grid. If your grid environment uses Platform Suite for SAS, you can manage competing priorities and workloads three ways:

- Job slots. They let you control how many jobs can run concurrently on each machine in the grid. This enables you to tune the load that each machine in the grid can accept. For example, you can assign a higher number of job slots to higher-capacity machines, which specifies that those machines can process more jobs concurrently.
- Queues. They let you control when jobs can run and what computing resources are available to each job that is submitted to the queue. You can create queues based on factors such as job size or priority. You can also define job dispatch windows and run windows for each queue. When you submit a job to a particular queue, the queue settings determine when the job runs and what priority the job has compared to other jobs that have been submitted to the grid. You can also specify the number of job slots across the grid that a queue can use. By combining the job-slot specification on the hosts and queues, you can specify how work is distributed across the grid.
- Partitions. They let you specify where jobs are run on the grid. Partitions are defined and used by specifying resource names on hosts and using matching resource names on jobs. The resource names are specified on machines in the grid to indicate what type of job each machine should run. When you submit jobs to the grid, you can specify resource names to specify which machines should be used to process the job.

Specifying Job Slots for Machines

Platform LSF uses job slots to specify the number of processes that are allowed to run concurrently on a machine. A machine cannot run more concurrent jobs than it has job slots. The default number of job slots for a machine is the same as the number of processor cores in the machine.

However, you can configure more than one job slot for each processor core. For machines with fast processors, configuring two jobs slots for each processor core enables you to take advantage of the processors' speed.

To change the number of job slots on a grid node, follow these steps:

1. Log on to the computer that hosts the grid controller as the LSF Administrator (lsfadmin).
2. Open the file **`lsb.hosts`**, which is located in the directory *LSF-install-dir*`\conf\lsbatch\`*cluster-name*`\configdir`. This is the LSF batch configuration file. Locate the Host section of the file, which contains an entry for a **`default`** grid node.

```
Begin
Host
HOST_NAME MXJ    r1m     pg    ls    tmp  DISPATCH_WINDOW  #Keywords
default    !     ()      ()    ()     ()     ()            #Example
End Host
```

3. Edit this file to specify the maximum number of job slots for all nodes or for each node.

   - To specify the total number of job slots for all nodes, edit the line for the **`default`** node. Here is an example:

     ```
     Begin Host
     HOST_NAME MXJ    r1m     pg    ls    tmp  DISPATCH_WINDOW  #Keywords
     default    !     ()      ()    ()     ()     ()            #Example
     End Host
     ```

     The value ! represents one job per processor for each node in the grid. You can replace this value with a number that represents the maximum number of job slots on each node.

   - To specify the total number of job per node, add a line for each node in the grid. Here is an example:

     ```
     Begin
     Host
     HOST_NAME MXJ    r1m     pg    ls    tmp  DISPATCH_WINDOW  #Keywords
     default    !     ()      ()    ()     ()     ()            #Example
     D1234      2     ()      ()    ()     ()     ()            #Example
     D1235      2     ()      ()    ()     ()     ()            #Example
     D1236      2     ()      ()    ()     ()     ()            #Example
     D1237      2     ()      ()    ()     ()     ()            #Example
     D1238      2     ()      ()    ()     ()     ()            #Example
     End Host
     ```

     Each line designates the concurrent execution of two jobs on each node.

4. Save and close the file.
5. Verify the LSF batch configuration file by entering this command at the command prompt: **`badmin reconfig`**
6. For details about using this command, see *Platform LSF Reference*.

Using Queues

Understanding Queues

When a job is submitted for processing on a grid that uses Platform Suite for SAS, it is placed in a queue and is held until resources are available for the job. LSF processes the jobs in the queues based on parameters in the queue definitions that establish criteria such as which jobs are processed first, what hosts can process a job, and when a job can be processed. All jobs submitted to the same queue share the same scheduling and control policy. By using multiple queues, you can control the workflow of jobs that are processed on the grid.

By default, SAS uses a queue named NORMAL. To use another queue that is already defined in the LSB.QUEUES file, specify the queue using a queue=*queue_name* option. You can specify this option either in the metadata for the SAS logical grid server (in the **Additional Options** field), or in the job options macro variable referenced in the GRDSVC_ENABLE statement. For information about specifying a queue in the logical grid server metadata, see "Modifying SAS Logical Grid Server Definitions" on page 17.

For information about specifying a queue in a GRDSVC_ENABLE statement, see "GRDSVC_ENABLE Function" on page 67.

Configuring Queues

Queues are defined in the LSB.QUEUES file, which is located in the directory *LSF-install-dir*`\conf\lsbatch\`*cluster-name*`\configdir`. The file contains an entry for each defined queue. Each entry names and describes the queue and contains parameters that specify the queue's priority and the attributes associated with the queue. For a complete list of parameters allowed in the lsb.queues file, refer to *Platform LSF Reference*.

Using the Normal Queue

As installed, SAS Grid Manager uses a default queue called NORMAL. If you do not specify the use of a different queue, all jobs are routed to this queue and are processed with the same priority. Other queues allow you to use priorities to control the work on the queues. The queue definition for a normal queue looks like the following:

```
Begin Queue
QUEUE_NAME = normal
PRIORITY = 30
DESCRIPTION = default queue
End Queue
```

Example: A High-Priority Queue

This example shows the existing queue for high priority jobs. Any jobs in the high-priority queue are sent to the grid for execution before jobs in the normal queue. The relative priorities are set by specifying a higher value for the PRIORITY attribute on the high priority queue.

```
Begin Queue
QUEUE_NAME = normal
PRIORITY = 30
DESCRIPTION = default queue
End Queue

Begin Queue
QUEUE_NAME = priority
PRIORITY = 40
DESCRIPTION = high priority users
End Queue
```

Example: A Night Queue

This example shows the existing queue for processing jobs (such as batch jobs) at night. The queue uses these features:

- The DISPATCH_WINDOW parameter specifies that jobs are sent to the grid for processing only between the hours of 6:00 PM and 7:30 AM.
- The RUN_WINDOW parameter specifies that jobs from this queue can run only between 6:00 PM and 8:00 AM. Any job that has not completed by 8:00 AM is suspended and resumed the next day at 6:00 PM.

- The HOSTS parameter specifies that all hosts on the grid except for host1 can run jobs from this queue. Because the queue uses the same priority as the normal queue, jobs from the high-priority queue still take precedence. Excluding host1 from the hosts available for the night queue leaves one host always available for processing jobs from other queues:

```
Begin Queue
QUEUE_NAME = normal
PRIORITY = 30
DESCRIPTION = default queue
End Queue

Begin Queue
QUEUE_NAME = priority
PRIORITY = 40
DESCRIPTION = high priority users
End Queue

Begin Queue
QUEUE_NAME = night
PRIORITY = 30
DISPATCH_WINDOW  = (18:00-07:30)
RUN_WINDOW = (18:00-08:00)
HOSTS = all ~host1
DESCRIPTION = night time batch jobs
End
Queue
```

Example: A Queue for Short Jobs

This example shows the existing queue for jobs that need to preempt longer-running jobs. The PREEMPTION parameter specifies which queues can be preempted as well as the queues that take precedence. Adding a value of PREEMPTABLE[short] to the normal queue specifies that jobs from the normal queue can be preempted by jobs from the short queue. Using a value of PREEMPTIVE[normal] to the short queue specifies that jobs from the short queue can preempt jobs from the normal queue. Using a value for PRIORITY on the short queue ensures that the jobs process before jobs from the normal queue, but that the jobs from the priority queue still take precedence.

```
Begin Queue
QUEUE_NAME = normal
PRIORITY = 30
PREEMPTION = PREEMPTABLE[short]
DESCRIPTION = default queue
End Queue

Begin Queue
QUEUE_NAME = priority
PRIORITY = 40
DESCRIPTION = high priority users
End Queue

Begin Queue
QUEUE_NAME = short
PRIORITY = 35
```

```
PREEMPTION = PREEMPTIVE[normal]
DESCRIPTION = short duration jobs
End Queue
```

Specifying Job Slot Limits on a Queue

A job slot is a position on a grid node that can accept a single unit of work or SAS process. Each host has a specified number of available job slots. By default, each host is configured with a single job slot for each core on the machine, so a multi-processor machine could have multiple job slots. In addition, you might want to specify multiple job slots for a single fast processor. For information about specifying job slots for a host, see *Platform LSF Reference*.

You can also use a queue definition to control the number of job slots on the grid or on an individual host that are used by the jobs from a queue. The QJOB_LIMIT parameter specifies the maximum number of job slots on the grid that can be used by jobs from the queue. The HJOB_LIMIT parameter specifies the maximum number of job slots on any one host that can be used by the queue. The following example sets a limit of 60 job slots across the grid that can be used and a limit of 2 job slots on any host that can be used.

```
Begin Queue
QUEUE_NAME = normal
PRIORITY = 30
DESCRIPTION = default queue
QJOB_LIMIT = 60
HJOB_LIMIT = 2
End Queue
```

Partitioning the Grid

Overview

Partitions enable you to specify where jobs are run on the grid. One method for creating and using partitions is to define resource names on grid nodes and then specify those same resource names on jobs that are sent to the grid. The resource names specified on grid machines indicate the type of job each machine runs (for example, jobs from specified applications or high-priority jobs), so you can direct specific types of work to the nodes that are best suited for processing those jobs.

By default, when a job is sent to the grid, the name of the SAS application server is sent as a resource name along with the job. You can further specify the type of machine used to run a job by specifying the WORKLOAD= parameter on the GRDSVC_ENABLE call.

For example, assume that you have installed and configured a grid that uses the application server name of SASApp. You now want to specify that SAS Data Integration Studio jobs should run on certain machines in the grid. To make this happen, follow these steps:

1. Create a resource name of DI for SAS Data Integration Studio jobs. (DI is only an example; you can use any user-defined string.)
2. Assign the resource names DI and SASApp to the machines that you want to use for processing SAS Data Integration Studio jobs.
3. Add the value DI to the **Workload** field for the logical grid server definition.

4. In SAS Data Integration Studio, choose the workload named DI in the Loop Properties window. This specifies that the job is sent to the DI workload, which sends the job to one of the machines with SASApp as a resource name and DI as a resource name. If there are no grid servers with resource names that match the value on the job, the job is not processed.

Defining Resources

With SAS 9.2, SAS Grid Manager provides the **addresource** command to define hosts and resources. To use this command to specify resource names, follow these steps:

1. Log on to the grid control machine as the LSF administrator.
2. Issue the command **addresource -r <*resource_name*> -m <*machine_name*>**.

   For example, the command **addresource -r DI -m D1234** assigns the resource name DI to the machine D1234.

Specifying Resource Names Using GRDSVC_ENABLE

You can use the GRIDSVC_ENABLE function to specify resource names for jobs that run on the grid. Use the SERVER= option to specify the SAS application server and the WORKLOAD= option to specify resource requirements for jobs. For more information, see "GRDSVC_ENABLE Function" on page 67.

Specifying Resource Names Using the SAS Grid Manager Client Utility

You can specify resource names when submitting SAS programs to the grid using the SAS Grid Manager Client Utility. Use the -GRIDWORKLOAD option to specify a resource name for the job. For more information, see "SASGSUB Syntax: Submitting a Job" on page 79.

Specifying Resource Names in SAS Data Integration Studio

In order to specify the resource name for SAS Data Integration Studio jobs, you must complete these tasks:

- Add the resource name as an allowed value for the logical grid server to which you send jobs.
- Specify the workload that corresponds to the resource name in the loop transformation properties.

To add the resource name to the logical grid server metadata's Workload values, see "Modifying SAS Logical Grid Server Definitions" on page 17.

To specify the workload value in SAS Data Integration Studio, follow these steps:

1. On the SAS Data Integration Studio menu bar, select **Tools** ⇨ **Options**, and then select the **SAS Server** tab on the Options dialog box.
2. Select the SAS grid server in the **Server** field.
3. Select the workload to use for the submitted jobs in the **Grid workload specification** field.

Removing the Resource Name Requirement

If you have a floating grid license and do not define resources on any grid nodes, sending the SAS application server name as a required resource causes all jobs sent to the grid to fail. A floating grid license enables you to have a large number of grid resources available for use (300 cores, for example) but use SAS Grid Manager to limit the number of SAS processes that can run concurrently on the grid to a smaller number (for example, 175). In this environment, you can change the metadata definition of the grid server to not require a resource name. To change the definition, follow these steps:

1. In SAS Management Console, open the Server Manager plug-in and locate the logical server definition for one of the servers identified in the lsf.cluster file.
2. Expand the logical Grid Server node and select the Grid Server node.
3. Select **Properties** from the pop-up menu or the **File** menu.
4. In the Properties window, select the **Options** tab.
5. Select the check box **Do not require SAS Application Server name as a grid resource**.
6. Save and close the definition.
7. Repeat this process for all grid servers.

If you remove the SAS application server name as a required resource, you can partition the grid by directing jobs to a specific queue that you have defined to limit the hosts and jobs slots that can be used. To set up this form of partitioning, follow these steps:

1. Follow the preceding procedure to remove the SAS application server name as a required resource.
2. Do not specify a workload value on the server definition.
3. In the Additional Options field for the SAS Logical Grid Server definition, specify `queue=<*new_queue_name*>'.
4. Define a new queue *new_queue_name* in the lsb.queues file. Use the definition to limit the hosts and job slots.

Restarting Jobs

Overview

Using SAS Grid Manager, you can restart jobs that have been submitted to the grid. You can restart a job based on a lack of response from the server running to job or on the receipt of a specified return code.

The restart capability is available only if you are using the SAS Grid Manager Client Utility or are scheduling grid jobs. It is not available if you are using the grid to start remote SAS/CONNECT servers.

Restarting Unresponsive Jobs

You can automatically restart a job when the server on which the job is running stops responding. To use this capability:

- add checkpoints to the SAS programs you submit to the grid
- submit programs to the grid using the SAS Grid Manager Client Utility and specifying the GRIDRESTARTOK argument

If the host that is running the job becomes unresponsive, the program is automatically restarted at the last checkpoint.

Note: This function is available only if you are using Platform Suite for SAS as the grid middleware.

Restarting Jobs Based on Return Codes

You can set up a queue that automatically requeues and restarts any job that ends with a specified return code. In order to use this functionality, you must use Platform Suite for SAS as the grid middleware, and you must be using the SAS Grid Manager Client Utility. This utility is included with the second maintenance release for SAS 9.2.

To set up a queue for automatic restart, follow these steps:

1. Create a queue, including the REQUEUE_EXIT_VALUES: *return_code_a return_code_b ...return_code_n* option in the queue definition. The *return_code* values are the job exit codes that you want to filter. Any job that exits with one of the specified codes will be restarted.

   Note: If you specify a return_code value higher than 255, LSF uses the modulus of the value with 256. For example, if SAS returns an exit code of 999, LSF sees that value as (999 mod 256), or 231. Therefore, you must specify a value of 231 on REQUE_EXIT_VALUES.

2. Specify the queue you created in step 1, either by modifying a grid server definition or by specifying the -GRIDJOBOPTS option.

   To create or modify a grid server definition, use the Server Manager plug-in in SAS Management Console. To specify the queue, specify **"queue=<*name_of_requeue_queue*>"** in the Additional Options field of the server definition.

   To use -GRIDJOBOPTS, submit the job using the -GRIDJOBOPTS queue=*name_of_requeue_queue* option.

3. Submit the job to the requeue queue on the grid. You must use the SAS Grid Manager Client Utility to specify the -GRIDRESTARTOK option. Send the job to the requeue queue by using the server you specified in step 2.

Modifying Configuration Files with Platform RTM for SAS

You can use Platform RTM for SAS to modify the configuration files that define queues and resources on the grid. The Platform RTM download package contains documentation

on performing this task. However, if you use Platform RTM for SAS to change any configuration files, you cannot make any further changes to the files outside of Platform RTM for SAS. Changes in the configuration files are not synchronized with Platform RTM for SAS.

Chapter 4

Enabling SAS Applications to Run on a Grid

Overview of Grid Enabling

After you have configured your grid, you can configure your SAS applications and programs to take advantage of the grid capabilities. Some SAS applications require you to change only an option to take advantage of the grid; other applications require more extensive changes. You can also use the SAS Grid Manager Client Utility to submit jobs to the grid from an operating system command line.

Using SAS Display Manager with a SAS Grid

Overview

You can use SAS Display Manager as a client to submit SAS programs to the grid for execution, with the results of the execution returned to the local workstation. When you submit a SAS program from a SAS Display Manager client to execute on a grid, the program runs on a grid machine in a separate SAS session with its own unique work library. The SAS log and output of the grid execution are returned to the local workstation. You might need to perform additional actions in order to view data from the SAS Display Manager session that was created or modified by the program that ran on the grid. For example, modifications might be required in order to use the Explorer to browse SAS libraries that are modified by grid execution.

Submitting Jobs from the Program Editor to the Grid

The first step in integrating SAS processes with the grid is to get your SAS programs running on the grid.

In order to submit a SAS program to the grid,you must add a set of grid statements to the program. For programs submitted through the SAS Program Editor, you can save the statements to an external file and then specify a key definition that issues the statements. Submit the contents of the SAS Program Editor window to the grid, rather than to the local workstation.

Some of the examples in this topic use SAS/CONNECT statements (such as signon, rsubmit, and signoff). For detailed information about these statements, see *SAS/CONNECT User's Guide*.

To add grid statements to a program and submit the program to the grid, follow these steps:

1. Save these statements to an external file, referred to as *grid-statement-file* (for example, c:\gpre.sas):

```
%global count;
%macro gencount;
%if %bquote(&count) eq %then %do; %let count=1;%end;%else %let
count=%eval(&count+1);
%mend;
%gencount;
options
metaserver='metadata-server-address';
options metaport=metadata-server-port;
```

```
options metauser=username;
options metapass="password";
%let rc=%sysfunc(grdsvc_enable(grid&count, resource=SASApp));
 signon grid&count;
```

metadata-server-address is the machine name of the SAS Metadata Server and *metadata-server-port* is the port used to communicate with the metadata server.

2. Open the Keys window and specify the following for an available key (for example, F12):

```
gsubmit "%include
'grid-statement_file';";
rsubmit grid&count wait=no persist=no;
```

grid-statement-file is the path and filename of the file (for example, c:\gpre.sas) containing the grid statements.

3. Type or include a SAS program in the Program Editor window, and then press the key to assigned to the grid statements. The program is automatically submitted to the grid for processing. Your local machine is busy only until the program is submitted to the grid.

Using the same key to submit multiple jobs causes multiple jobs to be executed in parallel on the grid.

Viewing LOG and OUTPUT Lines from Grid Jobs

The example in "Submitting Jobs from the Program Editor to the Grid" on page 34 uses asynchronous rsubmits. This causes the results of the execution to be returned to the local log and output windows only after the entire program finishes execution on the grid. To cause the log and output lines to be displayed while the program is executing, delete the **`options noconnectwait;`** line in the program.

The rsubmit executes synchronously, and the returned log and output lines are displayed while the job is executing. This also results in the Client SAS session being busy until the entire grid job has completed. You cannot submit more code until the job completes.

Using the SAS Explorer Window to Browse Libraries

The Client SAS session and the grid SAS session are two separate instances of SAS. Any code or products needed to access data must be submitted and available on both the client machine as well as the grid nodes. Use the following steps to browse libraries from the SAS Explorer Window that are accessed and modified by jobs executing in the grid:

1. Define all of your SAS libraries within SAS metadata under your server context (for example, under SASApp).
2. Ensure that the following option is in the SAS invocation in the sasgrid script file used to start SAS on the grid nodes. This option should have been added by the SAS Deployment Wizard.

```
metaautoresources SASApp
```

SASApp is the name of your application server context.

3. Include this option on the Client SAS session invocation on the workstation.

```
metaautoresources SASApp
```

SASApp is the name of your application server context.

Note: If you are accessing data through any SAS/ACCESS product, you must license the SAS/ACCESS products on the SAS Client machine in order to be able to browse those libraries from the SAS Explorer. The SAS/ACCESS products must also be licensed on the grid nodes in order to enable the job to access data during execution.

Each SAS session executing on the grid is a unique session with a unique WORK library. In order to view the work libraries that are created on each of the grid nodes, you must add the following line after the signon statement in the code provided in "Submitting Jobs from the Program Editor to the Grid" on page 34:

```
libname workgrid slibref=work server=grid&count;
```

grid&count is the label used as the remote session ID in the signon statement.

Submitting Batch SAS Jobs to the Grid

Overview

SAS Grid Manager Client Utility is a command-line utility that enables users to submit SAS programs to a grid for processing. This utility allows a grid client to submit SAS programs to a grid without having SAS installed on the machine performing the submission. It also enables jobs to be processed on the grid without requiring that the client remain active. You can use the command to submit jobs to the grid, view job status, retrieve results, and terminate jobs.

Most of the options that are used by the SAS Grid Manager Client Utility are contained in the sasgsub.cfg file. This file is automatically created by the SAS Deployment Wizard. These options specify the information that the SAS Grid Manager Client Utility uses every time it runs.

Submitting Jobs Using the SAS Grid Manager Client Utility

To submit a SAS job to a grid using the SAS Grid Manager Client Utility, change to the <*configuration_directory*>/Applications/SASGridManagerClientUtility/<*version*> directory and issue the following command from a SAS command line:

SASGSUB -GRIDSUBMITPGM *sas-program-file*

The -GRIDSUBMITPGM option specifies the name and path of the SAS program that you want to submit to the grid.

In addition, you can specify other options that are passed to the grid or used when processing the job, including workload resource names. For a complete list of options, see "SASGSUB Syntax: Submitting a Job" on page 79.

Viewing Job Status Using the SAS Grid Manager Client Utility

After you submit a job to the grid, you might want to check the status of the job. To check the status of a job, change to the <*configuration_directory*>/Applications/SASGridManagerClientUtility/<*version*> directory and issue the following command from a command line:

SASGSUB -GRIDGETSTATUS [*job-ID* | ALL]

-GRIDGETSTATUS specifies the ID of the job you want to check, or ALL to check the status of all jobs submitted by your user ID. For a complete list of options, see "SASGSUB Syntax: Viewing Job Status" on page 84.

Note: For the job status to be reported correctly, make sure that the sasgrid command file is updated on all grid nodes. The file is updated by the installation process for the second maintenance release after SAS 9.2. See "Modifying the SASGRID Script File" on page 22 for more information.

The following is an example of the output produced by the SASGSUB -GRIDGETSTATUS command.

```
Current Job Information
  Job 1917 (testPgm) is Finished:  Submitted: 08Dec2008:10:28:57, Started: 08Dec2008:10:28:57 on Host
host1, Ended: 08Dec2008:10:28:57
  Job 1918 (testPgm) is Finished:  Submitted: 08Dec2008:10:28:57, Started: 08Dec2008:10:28:57 on Host
host1, Ended: 08Dec2008:10:28:57
  Job 1919 (testPgm) is Finished:  Submitted: 08Dec2008:10:28:57, Started: 08Dec2008:10:28:57 on Host
host1, Ended: 08Dec2008:10:28:57
  Job information in directory U:\pp\GridSub\GridWork\user1\SASGSUB-2008-11-24_13.17.17.327_testPgm is
invalid.
  Job 1925 (testPgm) is Submitted: Submitted: 08Dec2008:10:28:57
```

Ending Jobs Using the SAS Grid Manager Client Utility

If a job that has been submitted to the grid is causing problems or otherwise needs to be terminated, use the SAS Grid Manager Client Utility to end the job. Change to the <*configuration_directory*>/Applications/SASGridManagerClientUtility/<*version*> directory and issue the following command from a command line:

SASGSUB -GRIDKILLJOB [*job-ID* | ALL]

-GRIDKILLJOB specifies the ID of the job you want to end, or ALL to end all jobs submitted by your user ID. For a complete list of options, see "SASGSUB Syntax: Ending a Job" on page 82.

Retrieving Job Output Using the SAS Grid Manager Client Utility

After a submitted job is complete, use the SAS Grid Manager Client Utility to retrieve the output produced by the job. Change to the <*configuration_directory*>/Applications/SASGridManagerClientUtility/<*version*> directory and issue the following command from a command line:

SASGSUB -GRIDGETRESULTS [*job-ID* | ALL] [-GRIDRESULTSDIR *directory*]

-GRIDGETRESULTS specifies the ID of the job whose results you want to retrieve, or you can specify ALL to retrieve the results from all jobs submitted by your user ID.

-GRIDRESULTSDIR specifies the directory in which the jobs results should be moved. When the results are retrieved, they are removed from the GRIDWORK directory, which keeps this directory from filling up with completed jobs.

A file named job.info is created along with the job output. This file contains information about the execution of the job, including the submit time, start time, and end time, the machine on which the job ran, and the job ID.

The following is an example of the output produced by the SASGSUB -GRIDGETRESULTS command.

```
Current Job Information
  Job 1917 (testPgm) is Finished:  Submitted: 08Dec2008:10:53:33, Started: 08Dec2008:10:53:33 on Host
host1, Ended: 08Dec2008:10:53:33
    Moved job information to .\SASGSUB-2008-11-21_21.52.57.130_testPgm

  Job 1918 (testPgm) is Finished:  Submitted: 08Dec2008:10:53:33, Started: 08Dec2008:10:53:33 on Host
host1, Ended: 08Dec2008:10:53:33
    Moved job information to .\SASGSUB-2008-11-24_13.13.39.167_testPgm

  Job 1919 (testPgm) is Finished:  Submitted: 08Dec2008:10:53:34, Started: 08Dec2008:10:53:34 on Host
host1, Ended: 08Dec2008:10:53:34
    Moved job information to .\SASGSUB-2008-11-24_13.16.06.060_testPgm

  Job 1925 (testPgm) is Submitted: Submitted: 08Dec2008:10:53:34
```

Locating Grid Provider Files

If you are using a grid provider other than Platform Suite for SAS, you might need to specify the location of the grid provider's Java files before you can submit jobs to the grid. You can specify the location by using the -GRIDPLUGINPATH option for the SAS Grid Manager Client Utility, either as a command-line option or in the sasgsub.cfg file.

Scheduling Jobs on a Grid

Using the scheduling capabilities, you can specify that jobs are submitted to the grid when a certain time has been reached or after a specified file or job event has occurred (such as a specified file being created).

To schedule a job to run on a grid, follow these steps:

1. Deploy the job for scheduling.

   Some SAS applications, such as SAS Data Integration Studio, include an option to deploy jobs for scheduling. If you want to schedule an existing SAS job, use the **Deploy SAS DATA Step Program** option in the Schedule Manager plug-in of SAS Management Console.

2. Use the Schedule Manager plug-in in SAS Management Console to add the job to a flow.

   A flow contains one or more deployed jobs as well as the schedule information and time, file, or job events that determine when the job runs.

3. Assign the flow to a scheduling server and submit the flow for scheduling.

   You must assign the flow to a Platform Process Manager scheduling server in order for the scheduled job to run on the grid.

For detailed information about scheduling, see *Scheduling in SAS*.

Comparing Grid Submission Methods

You can use the SAS Grid Manager Client Utility, the Schedule Manager plug-in to SAS Management Console, and SAS language statements to submit jobs to the grid. The following table compares the methods.

Feature	SAS Grid Manager Client Utility	Schedule Manager Plug-In	SAS Language Statements
Interface	Command line	SAS Management Console interface	SAS language syntax
Duration of client connection	Duration of the submission	Duration of the submission	Duration of the execution
Minimum client installation requirements	SAS Grid Manager Client Utility and Platform LSF	SAS Management Console	Base SAS, SAS/CONNECT, Platform LSF
Support for checkpoint restart	Yes	Yes	No
Support for SAS options, grid options, and policies	Yes	Yes	Yes
Support for event-triggered workflow execution	No	Yes	No
Support for all grid providers supported by SAS	Yes	No	Yes

Enabling Distributed Parallel Execution of SAS Jobs

Some SAS programs contain multiple independent subtasks that can be distributed across the grid and executed in parallel. This approach enables the application to run faster. To enable a SAS program to use distributed parallel processing, add RSUBMIT and ENDRSUBMIT statements around each subtask and add the GRDSVC_ENABLE function call. The SAS Grid Manager automatically assigns each identified subtask to a grid node.

You can use the SAS Code Analyzer to automatically create a grid-enabled SAS job. To use the SAS Code Analyzer, add PROC SCAPROC statements to your SAS program, specifying the GRID parameter. When you run the program with the PROC SCAPROC statements, the grid-enabled job is saved to a file. You can then run the saved SAS job on the grid, and the SAS Grid Manager automatically assigns the identified subtasks to a grid node.

An example of the syntax for the SAS Code Analyzer is:

```
proc scaproc;
     record '1.txt' grid '1.grid':
run;
remainder of SAS program...
```

For complete information and syntax for the PROC SCAPROC statement, see *Base SAS Procedures Guide*.

An example of the syntax used for enabling distributed parallel processing is:

```
% let rc=%sysfunc(grdsvc_enable(_all_,
resource=SASApp));
options autosignon;
rsubmit task1 wait=no;
    /* code for parallel task #1  */
endrsubmit;
rsubmit task2 wait=no;
    /* code for parallel task #2 */
endrsubmit;
. . .
rsubmit taskn wait=no;
     /* code for parallel task #n   */
endrsubmit;
waitfor _all_ task1 task2 . . . taskn;
signoff _all_;
```

For more information, see "GRDSVC_ENABLE Function" on page 67.

For detailed syntax information, see *SAS/CONNECT User's Guide*.

Using SAS Enterprise Guide with a SAS Grid

Submitting SAS Programs to the Grid Using SAS Enterprise Guide

SAS Enterprise Guide provides an option to automatically add the necessary grid statements to all submitted programs or tasks. To run programs submitted from SAS Enterprise Guide on the grid, follow these steps:

1. In SAS Enterprise Guide, select **Tools** ⇨ **Options** to open the Options window.
2. In the Options window, select **SAS Programs**. To enable SAS Enterprise Guide tasks to run on a SAS grid, select **Tasks** ⇨ **Custom Code** instead.
3. Select **Insert custom SAS code before submitted code** and then click **Edit**.
4. In the Edit window, enter these SAS statements:

   ```
   options metaserver='metadata-server-address';
   options metaport=metadata-server-port;;
   %let rc=%sysfunc(grdsvc_enable(_all_,resource=SASApp));
   signon task1;
   rsubmit;
   ```

5. In the Options window, select **Insert custom SAS code after submitted code**, and then click **Edit**.
6. In the Edit window, enter these SAS statements:

```
endrsubmit;
signoff;
```

7. While testing, if you want to verify that the program ran on the grid, include this statement before the **signoff** statement:

   ```
   %put This code ran on the machine %sysfunc(grdsvc_getname(task1));
   ```

   You should remove this statement when the code runs in a production environment.

Alternatively, you can run SAS Enterprise Guide jobs on a grid through workspace server load balancing. After you set up a workspace server to use load balancing, you can submit SAS Enterprise Guide jobs to the server to automatically use the load balancing capability. See "Using SAS Grid Manager for Workspace Server Load Balancing" on page 49.

Generating ODS Output on the Grid Using SAS Enterprise Guide

You can specify options for the results of SAS programs or tasks that are run by SAS Enterprise Guide. If you are running these programs and tasks on a grid, you must propagate these settings to all grid nodes so that the output from the nodes is formatted properly. To apply the result settings to all grid nodes, follow these steps:

Note: This procedure requires either SAS Enterprise Guide Version 4.22 or Version 4.1 with hotfix 11 (41EG11) applied.

1. In SAS Enterprise Guide, select **Tools** ⇨ **Options** ⇨ **Results** to specify the result options.
2. In the SAS Enterprise Guide Options window, select **Results** ⇨ **Results General**. Uncheck the **Link handcoded ODS results** check box.

   This option enables the temporary files that are used by the grid sessions to be copied to the local SAS Enterprise Guide project directories.
3. Close the Options window.
4. Edit the SAS\Enterprise Guide 4\SEGuide.exe.config file and add this line:

   ```
   <add key="OdsOptionsToMacro" value="true" />
   ```

   This statement causes SAS Enterprise Guide to generate macro statements for the results options that you specified.
5. (Optional) If all programs and tasks submitted from SAS Enterprise Guide will run on the grid, you can add a statement to suppress the ODS statements for the SAS Workspace Server. This statement eliminates all of the default ODS result entries in the workspace and forces the programs to use the settings that are in place on the grid nodes.

   Add this line to the SAS\Enterprise Guide 4\SEGuide.exe.config file:

   ```
   <add key="SuppressODSStetements" value="true" />
   ```
6. Restart SAS Enterprise Guide.

After the change is applied, SAS Enterprise Guide applies the result option settings to a set of macros. For example, if HTML is set as the only result output, the macro statements will look like this:

```
/* BEGIN: SAS Enterprise Guide results options */
%LET _GOPT_DEVICE = ACTIVEX;
%LET _GOPT_XPIXELS = 0;
%LET _GOPT_YPIXELS = 0;
```

```
%LET _GOPT_GFOOTNOTE = NOGFOOTNOTE;
%LET _GOPT_GTITLE = NOGTITLE;
%LET _ODSOPTIONS_GRAPHCODEBASE = ATTRIBUTES=("CODEBASE"="http://www2.sas.com/
   codebase/graph/v91/sasgraph.exe");
%LET _ODSDEST_LISTING = ;
%LET _ODSDEST_HTML = HTML;
%LET _ENCODING_HTML = utf-8;
%LET _ODSSTYLE_HTML = Analysis;
%LET _ODSSTYLESHEET_HTML = (URL="http://support.sas.com/styles/analysis.css");
%LET _ODSDEST_RTF = ;
%LET _ODSDEST_PDF = ;
%LET _ODSDEST_SRX = ;
/* END: SAS Enterprise Guide results options */
```

You can then add macros to the grid wrapper code to evaluate the active preferences and propagate the appropriate settings to the grid session. The wrapper code for the macros listed previously looks like the following:

```
options metaserver='server1.domain.com';
options metaport=8561;
%let rc=%sysfunc(grdsvc_enable(_all_,resource=SASMain));
signon task1;
%include "c:\htmllocal.sas" ;
%include "c:\rtflocal.sas" ;
%include "c:\pdflocal.sas" ;
%include "c:\srxlocal.sas" ;
rsubmit;
ODS _ALL_ CLOSE;
%inc "c:\htmlremote.sas" ;
%inc "c:\rtfremote.sas" ;
%inc "c:\pdfremote.sas" ;
%inc "c:\srxremote.sas" ;
```

The settings for each type of output are contained in a set of *local.sas macro files (such as htmllocal.sas). The files use the %SYSRPUT macro to propagate the settings to the grid session.

The ODS option statements are submitted through a set of *remote.sas macro files (such as htmlremote.sas).

Sample SEGuide.exe.config files (for both SAS Enterprise Guide 4.22 and 4.1) as well as sample macro wrappers and sample *local.sas and *remote.sas macro files are available at **http://support.sas.com/rnd/scalability/grid/download.html**.

Accessing Temporary Files Between Grid Nodes

SAS Enterprise Guide stores output data in the SASUSER library on the SAS Workspace Server machine, or in the EGTASK library if that library is defined. When a job or task from SAS Enterprise Guide runs on a grid, there are temporary work files that might need to be accessed between the grid nodes. In order for the multiple SAS grid sessions to be able to access these files, you must define a permanent shared library.

To create a permanent shared library for SAS Enterprise Guide jobs, use one of these methods:

- Use SAS Management Console to define the EGTASK library, pointing it to a shared storage location. Mark the library definition as Pre-assigned so that it is defined each

time a grid session is started. If you use this method, you have to change only one library definition if you want to change the storage location.

- Add an environment variable to the sasgrid.cmd (Windows) or sasgrid (UNIX) file that defines the EGTASK LIBNAME, pointing the library to a shared location. If you run the sasgrid.cmd or sasgrid file from a shared location, you have to change the LIBNAME definition statement only once if you want to change the library's location.
- Add the EGTASK LIBNAME statement to the autoexec file, pointing the library to a shared location. After the LIBNAME statement is added, add the -AUTOEXEC option to the command used to start SAS in the sasgrid.cmd (Windows) or sasgrid (UNIX).

Making SASWORK Libraries Visible to SAS Enterprise Guide

If you want the SASWORK libraries created by the grid sessions to be visible in the Library window of SAS Enterprise Guide, add a LIBNAME statement after the rsubmit statement in the grid wrapper code. For example, add this statement:

```
libname work1 (work);
```

The Library window will display the WORK1 library. You can then use this library window to display the contents of the library.

Assigning SAS Enterprise Guide Libraries in a Grid

In SAS 9.2 and later versions, SAS sessions on the grid use the METAAUTORESOURCES option by default. This option causes SAS libraries that are defined in metadata and identified as "pre-assigned" to automatically be assigned when the SAS session is started. Using pre-assigned libraries with the METAAUTORESOURCES option ensures that the libraries used in the code generated by SAS Enterprise Guide are available to the SAS sessions on the grid.

However, if your programs use a large number of libraries, you might not want to make all of these libraries pre-assigned. Automatically assigning a large number of libraries could cause performance problems, and not all libraries are likely to be used for all programs. To minimize the performance overhead, define the libraries in SAS metadata but do not identify them as pre-assigned. When you need to refer to the library, you can then use a LIBNAME statement using the META LIBNAME engine.

Developing SAS Programs Interactively Using SAS Enterprise Guide and a Grid

Maintaining a Connection to the Grid

By default, when you start SAS Enterprise Guide, it connects to a single workspace server and keeps that connection active for the length of the session. If you interactively develop programs in SAS Enterprise Guide by highlighting and submitting lines of code, the codes uses items such as libraries, WORK files, and SAS global statements on the workspace server. If you are using SAS Enterprise Guide in a grid environment, the items such as libraries and SAS global statements must be accessed through the grid, rather than a single workspace server. To maintain access to these items, you must maintain a connection to the grid while you are developing programs interactively.

To keep the connection between SAS Enterprise Guide and the grid active, remove the **signoff** statement from the wrapper code that executes at the end of each SAS Enterprise Guide program or task submitted to the grid. See "Submitting SAS Programs to the Grid Using SAS Enterprise Guide" on page 40 for the statements in the wrapper code.

Setting Workload Values

When SAS Enterprise Guide is used for interactive program development, the workload is likely to consist of short bursts of work interspersed with varying periods of inactivity while the user considers their next action. The SAS grid configuration can best support this scenario with these configuration settings:

- Increase the number of job slots for each machine.

  Increasing the number of job slots increases the number of simultaneous SAS sessions on each grid node. Because the jobs that are run on the grid are not I/O or compute intensive like large batch jobs, more jobs can be run on each machine.

- Implement CPU utilization thresholds for each machine.

  If all users submit CPU-intensive work at the same time, SAS Grid Manager can suspend some jobs and resume the suspended jobs when resources are available. This capability prevents resources from being overloaded.

The following example shows a sample LSB.HOSTS file that is configured with job slots set to 12 and CPU utilization thresholds set to 80%. The settings needed for a specific site will depend on the number of users and the size of the grid nodes.

```
HOST_NAME MXJ    ut        r1m     pg    ls    tmp  DISPATCH_WINDOW  #Keywords
#default   !     ()         ()     ()    ()     ()     ()             #Example
host01     12    0.7/0/8    ()     ()    ()     ()     ()             # host01
host02     12    0.7/0/8    ()     ()    ()     ()     ()             # host02
host03     12    0.7/0/8    ()     ()    ()     ()     ()             # host03
host04     12    0.7/0/8    ()     ()    ()     ()     ()             # host04
host05     12    0.7/0/8    ()     ()    ()     ()     ()             # host05
End Host
```

Using SAS Data Integration Studio with a SAS Grid

Scheduling SAS Data Integration Studio Jobs on a Grid

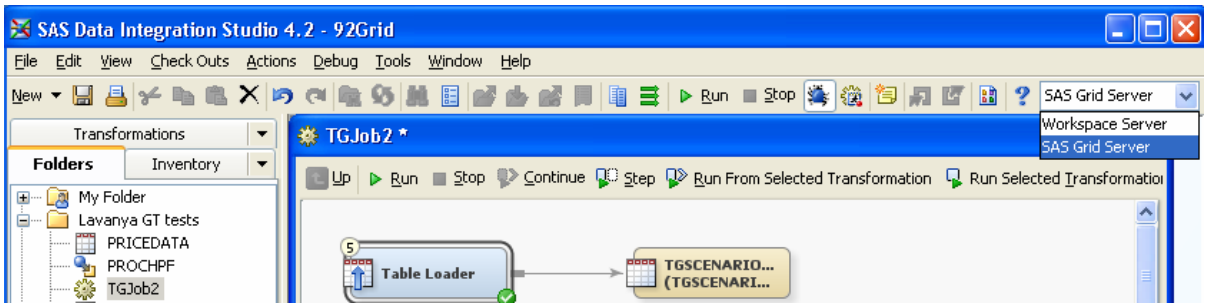
If your SAS grid environment uses Platform Suite for SAS as a middleware provider, you can schedule jobs from within SAS Data Integration Studio and have those jobs run on the grid. You deploy the job for scheduling in SAS Data Integration Studio, and then use the Schedule Manager plug-in in SAS Management Console to specify the schedule and the scheduling server. For more information, see For more information, see "Scheduling Jobs on a Grid" on page 38 . Also see *Scheduling in SAS*. .

Multi-User Workload Balancing with SAS Data Integration Studio

SAS Data Integration Studio 4.2 enables users to directly submit jobs to a grid. This capability allows the submitted jobs to take advantage of load balancing and job prioritization that you have specified in your grid. SAS Data Integration Studio also enables you to specify the workload that submitted jobs should use. This capability enables users to submit jobs to the correct grid partition for their work.

To submit a job to the grid, select the SAS Grid Server component in the **Server** menu on the Job Editor toolbar. Click **Submit** in the toolbar to submit the job to the grid.

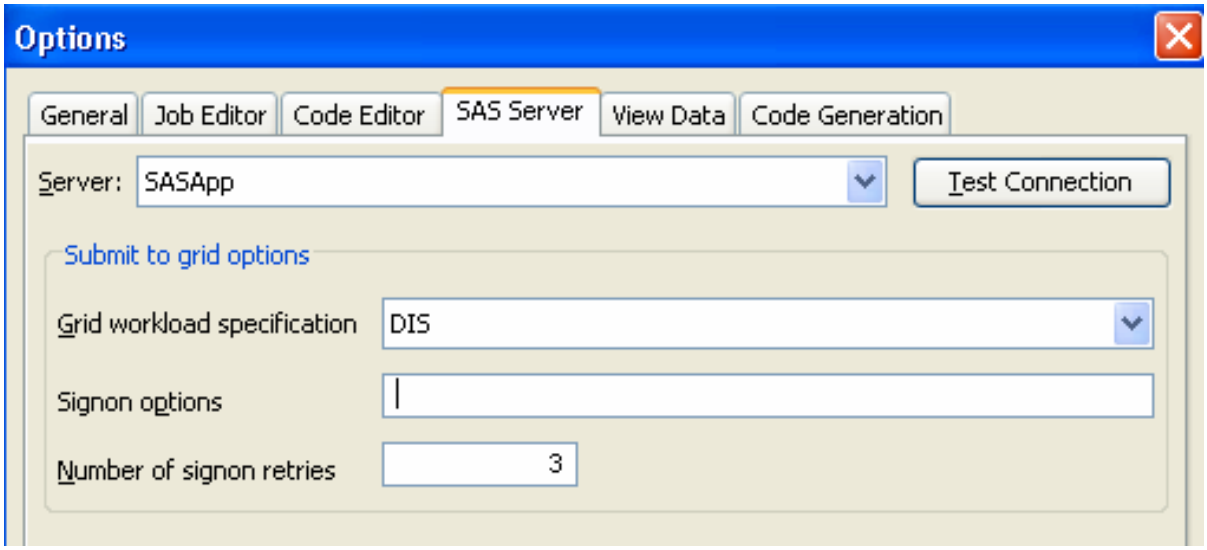
Display 4.1 *Submitting a Job to the Grid*



To specify a workload value for the server, follow these steps:

1. On the SAS Data Integration Studio menu bar, select **Tools** ⇨ **Options**, and then select the **SAS Server** tab on the Options dialog box.
2. Select the SAS grid server in the **Server** field.
3. Select the workload to use for the submitted jobs in the **Grid workload specification** field.

Display 4.2 *Selecting the Workload*

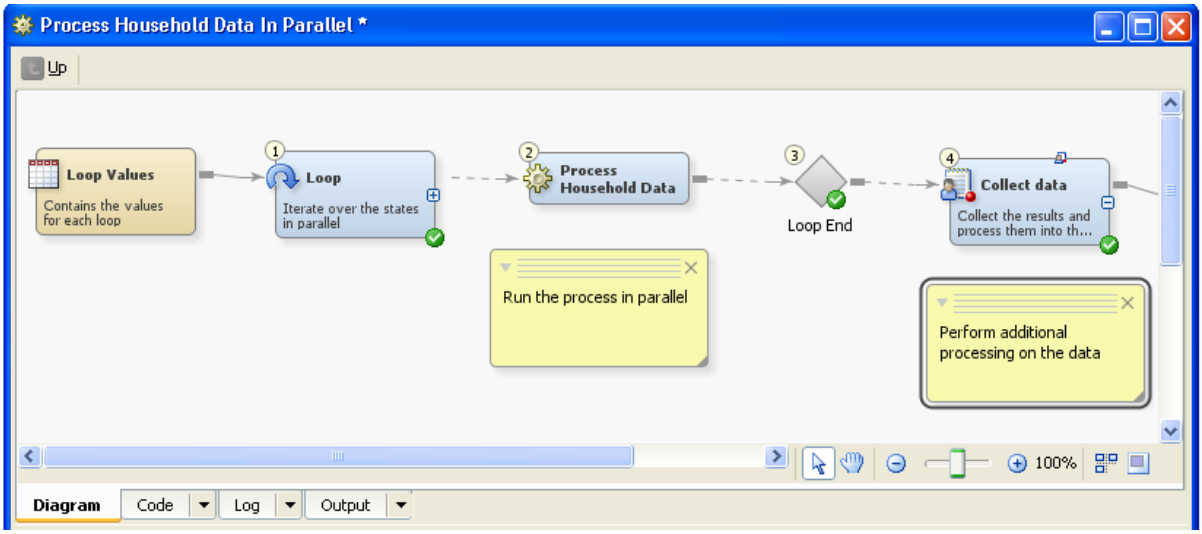


SAS Grid Manager uses the workload value to send the submitted job to the appropriate grid partition. For more information about the other steps required, see "Partitioning the Grid " on page 28.

Parallel Workload Balancing with SAS Data Integration Studio

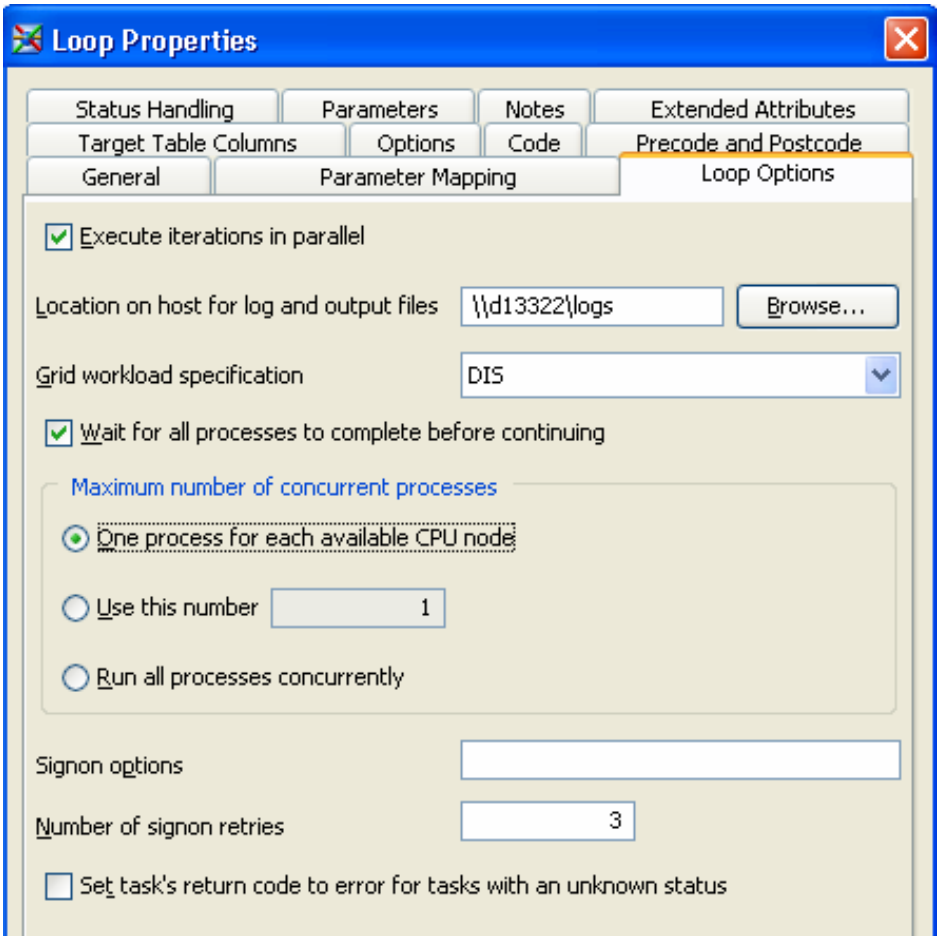
A common workflow in applications created by SAS Data Integration Studio is to repeatedly execute the same analysis against different subsets of data. Rather than running the process against each table in sequence, use a SAS grid environment to run the same process in parallel against each source table, with the processes distributed among the grid nodes. For this workflow, the Loop and Loop-End transformation nodes can be used in SAS Data Integration Studio to automatically generate a SAS application that spawns each iteration of the loop to a SAS grid via SAS Grid Manager.

Display 4.3 *Loop and Loop-End Transformation Nodes*



To specify options for loop processing, open the Loop Properties window and select the Loop Options tab. You can specify the workload for the job, as well as how many processes can be active at once.

Display 4.4 *Loop Properties Dialog Box*



For more information, see *SAS Data Integration Studio: User's Guide*.

Updating SAS Grid Server Definitions for Partitioning

After defining resource names, you can update the grid server metadata so that SAS Data Integration Studio knows the available resource names. To update the definitions, follow these steps:

1. In SAS Management Console, open the Server Manager plug-in and locate the logical server definition.
2. Expand the logical Grid Server node and select the Grid Server node. Select Properties from the pop-up menu or the File menu.
3. In the Properties window, select the **Options** tab.
4. Specify the workload resource name (for example, DI) in the **Workload** field.
5. Save and close the definition.
6. Repeat this process for all workloads.

Specifying Workload for the Loop Transformation

A SAS Data Integration Studio user performs these steps to specify an LSF resource in the properties for a Loop Transformation in a SAS Data Integration Studio job. When the job is submitted for execution, it is submitted to one or more grid nodes that are associated with the resource.

It is assumed that the default SAS application server for SAS Data Integration Studio has a Logical SAS Grid Server component, which was updated in the metadata repository. For more information, see "Partitioning the Grid " on page 28.

1. In SAS Data Integration Studio, open the job that contains the Loop Transformation to be updated.
2. In the Process Designer window, right-click the metadata object for the Loop Transformation and select **Properties**.
3. In the Properties window, click the **Loop Options** tab.
4. On the **Loop Options** tab, in the **Grid workload specification** text box, enter the name of the desired workload, such as DI. The entry is case sensitive.
5. Click **OK** to save your changes, and close the Properties window.

Using SAS Enterprise Miner with a SAS Grid

There are three cases where SAS Enterprise Miner uses a SAS grid:

- during model training, for parallel execution of nodes within a model training flow
- during model training, for load balancing of multiple flows from multiple data modelers
- during model scoring, for parallel batch scoring

The workflow for SAS Enterprise Miner during the model training phase consists of executing a series of different models against a common set of data. Model training is CPU-

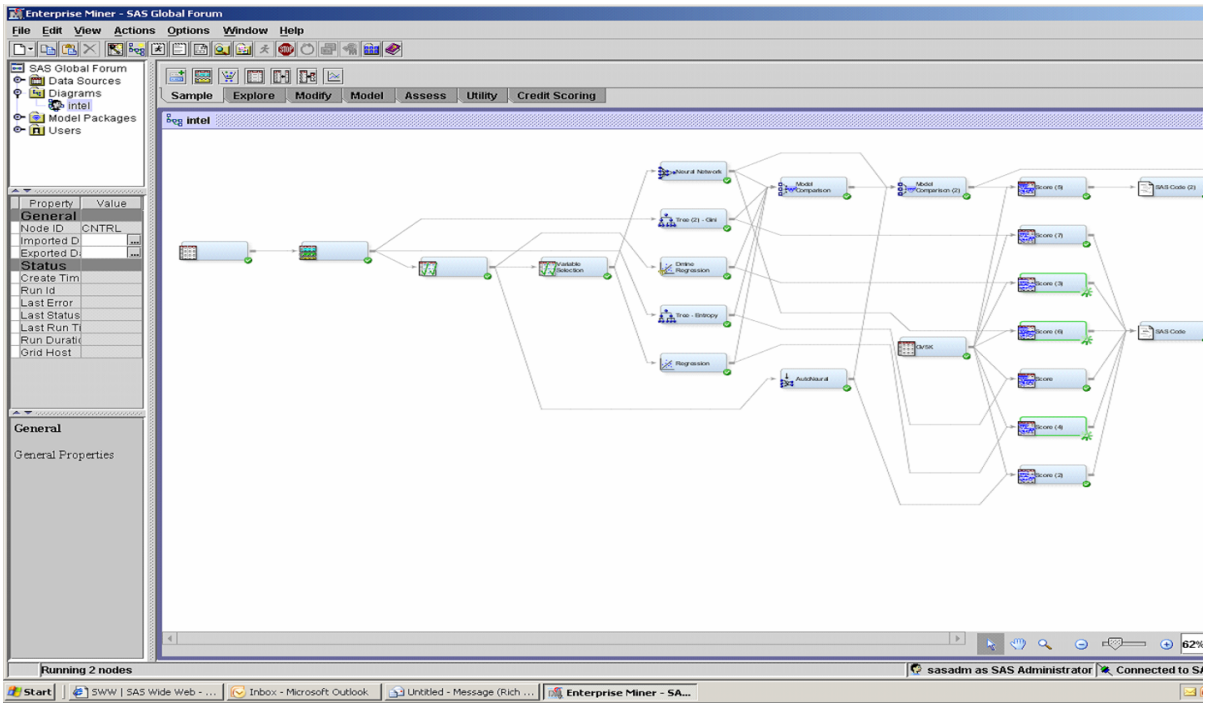
and I/O-intensive. The process flow diagram design of SAS Enterprise Miner lends itself to processing on a SAS grid, because each model is independent of the other models. SAS Enterprise Miner generates the SAS program to execute the user-created flow, and also automatically inserts the syntax needed to run each model on the grid. Because the models can execute in parallel on the grid, the entire process is accelerated.

In addition, SAS Enterprise Miner is typically used by multiple users who are simultaneously performing model training. Using a SAS grid can provide multi-user load balancing of the flows that are submitted by these users, regardless of whether the flows contain parallel subtasks.

The output from training a model is usually Base SAS code that is known as scoring code. The scoring code is a model, and there are usually many models that need to be scored. You can use SAS Grid Manager to score these models in parallel. This action accelerates the scoring process. You can use any of these methods to perform parallel scoring:

- Use the grid wrapper code to submit each model independently to the SAS grid.
- Use the Schedule Manager plug-in to create a flow that contains multiple models and schedule the flow to the SAS grid. Because each model is independent, they are distributed across the grid when the flow runs.
- Use SAS Data Integration Studio to create a flow to loop multiple models, which spawns each model to the SAS grid.

Display 4.5 *Grid Processing with SAS Enterprise Miner*



Using SAS Risk Dimensions with a SAS Grid

The iterative workflow in SAS Risk Dimensions is similar to that in SAS Data Integration Studio. Both execute the same analysis over different subsets of data. In SAS Risk Dimensions, the data is subsetted based on market states or by instruments. Each iteration

of the analysis can be submitted to the grid using SAS Grid Manager to provide load balancing and efficient resource allocation.

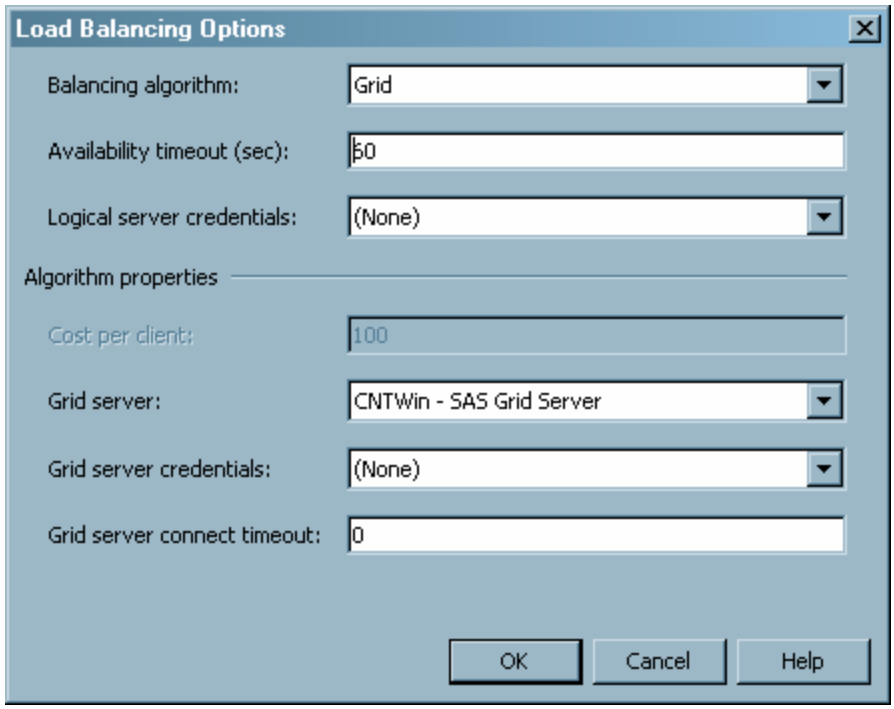
Because every implementation is different, an implementation of SAS Risk Dimensions in a grid environment must be customized to your specific business and data requirements.

Using SAS Grid Manager for Workspace Server Load Balancing

SAS Grid Manager can provide load balancing capabilities for SAS Workspace Servers. After you convert a SAS Workspace Server to use grid load balancing, the SAS Grid Manager examines any request for work that is sent to the workspace server and then determines which server in a cluster of workspace servers should process the job. This configuration provides server-side load balancing for any SAS product or solution that uses a SAS Workspace Server for processing. Products that can use this configuration include SAS Enterprise Guide, SAS Data Integration Studio, SAS Enterprise Miner, and SAS Marketing Automation.

To use the SAS Grid Manager for load balancing, follow these steps:

1. In the Server Manager plug-in in SAS Management Console, select the Logical Workspace Server that is to use load balancing.
2. Select **Convert To** ⇨ **Load Balancing** from the Actions menu or the context menu. The Load Balancing Options dialog box is displayed.



3. Specify the following values:

   Balancing algorithm
   Select **Grid**.

   Grid server
   Select a Grid Server that was defined during installation and configuration.

Grid server credentials
Select the credentials used to connect to the grid server.

4. Click OK to save your changes to the SAS Workspace Server metadata.

Chapter 5

Using Grid Management Applications

Using Grid Manager Plug-in

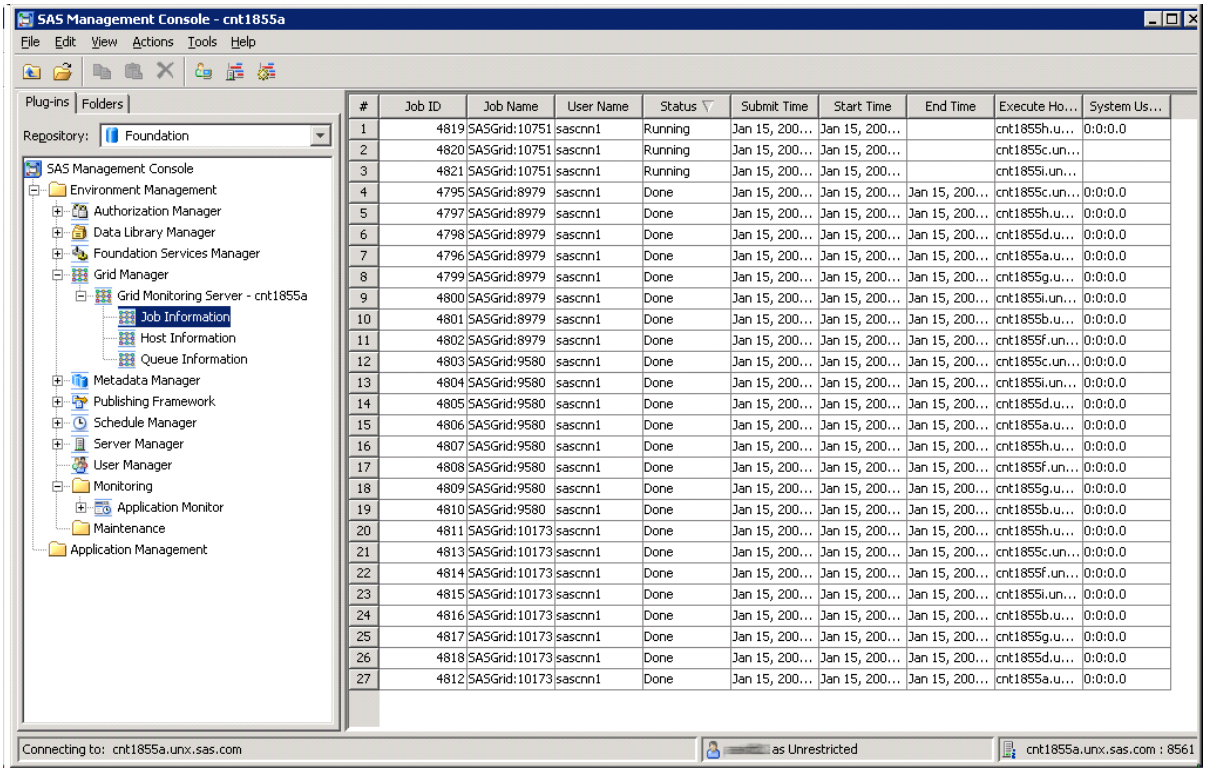
Overview

The Grid Manager plug-in for SAS Management Console enables you to monitor SAS execution in a grid environment. This plug-in enables you to manage workloads on the grid by providing dynamic information about the following:

- jobs that are running on the grid
- computers that are configured in the grid
- job queues that are configured in the grid

Information is displayed in tabular or chart format. Here is an example of a job view:

Display 5.1 *Job View in Grid Manager Plug-In to SAS Management Console*



#	Job ID	Job Name	User Name	Status	Submit Time	Start Time	End Time	Execute Ho...	System Us...
1	4819	SASGrid:10751	sascnn1	Running	Jan 15, 200...	Jan 15, 200...		cnt1855h.u...	0:0:0.0
2	4820	SASGrid:10751	sascnn1	Running	Jan 15, 200...	Jan 15, 200...		cnt1855c.un...	
3	4821	SASGrid:10751	sascnn1	Running	Jan 15, 200...	Jan 15, 200...		cnt1855i.un...	
4	4795	SASGrid:8979	sascnn1	Done	Jan 15, 200...	Jan 15, 200...	Jan 15, 200...	cnt1855c.un...	0:0:0.0
5	4797	SASGrid:8979	sascnn1	Done	Jan 15, 200...	Jan 15, 200...	Jan 15, 200...	cnt1855h.u...	0:0:0.0
6	4798	SASGrid:8979	sascnn1	Done	Jan 15, 200...	Jan 15, 200...	Jan 15, 200...	cnt1855d.u...	0:0:0.0
7	4796	SASGrid:8979	sascnn1	Done	Jan 15, 200...	Jan 15, 200...	Jan 15, 200...	cnt1855a.u...	0:0:0.0
8	4799	SASGrid:8979	sascnn1	Done	Jan 15, 200...	Jan 15, 200...	Jan 15, 200...	cnt1855g.u...	0:0:0.0
9	4800	SASGrid:8979	sascnn1	Done	Jan 15, 200...	Jan 15, 200...	Jan 15, 200...	cnt1855i.un...	0:0:0.0
10	4801	SASGrid:8979	sascnn1	Done	Jan 15, 200...	Jan 15, 200...	Jan 15, 200...	cnt1855b.u...	0:0:0.0
11	4802	SASGrid:8979	sascnn1	Done	Jan 15, 200...	Jan 15, 200...	Jan 15, 200...	cnt1855f.un...	0:0:0.0
12	4803	SASGrid:9580	sascnn1	Done	Jan 15, 200...	Jan 15, 200...	Jan 15, 200...	cnt1855c.un...	0:0:0.0
13	4804	SASGrid:9580	sascnn1	Done	Jan 15, 200...	Jan 15, 200...	Jan 15, 200...	cnt1855i.un...	0:0:0.0
14	4805	SASGrid:9580	sascnn1	Done	Jan 15, 200...	Jan 15, 200...	Jan 15, 200...	cnt1855d.u...	0:0:0.0
15	4806	SASGrid:9580	sascnn1	Done	Jan 15, 200...	Jan 15, 200...	Jan 15, 200...	cnt1855a.u...	0:0:0.0
16	4807	SASGrid:9580	sascnn1	Done	Jan 15, 200...	Jan 15, 200...	Jan 15, 200...	cnt1855h.u...	0:0:0.0
17	4808	SASGrid:9580	sascnn1	Done	Jan 15, 200...	Jan 15, 200...	Jan 15, 200...	cnt1855f.un...	0:0:0.0
18	4809	SASGrid:9580	sascnn1	Done	Jan 15, 200...	Jan 15, 200...	Jan 15, 200...	cnt1855g.u...	0:0:0.0
19	4810	SASGrid:9580	sascnn1	Done	Jan 15, 200...	Jan 15, 200...	Jan 15, 200...	cnt1855b.u...	0:0:0.0
20	4811	SASGrid:10173	sascnn1	Done	Jan 15, 200...	Jan 15, 200...	Jan 15, 200...	cnt1855h.u...	0:0:0.0
21	4813	SASGrid:10173	sascnn1	Done	Jan 15, 200...	Jan 15, 200...	Jan 15, 200...	cnt1855c.un...	0:0:0.0
22	4814	SASGrid:10173	sascnn1	Done	Jan 15, 200...	Jan 15, 200...	Jan 15, 200...	cnt1855f.un...	0:0:0.0
23	4815	SASGrid:10173	sascnn1	Done	Jan 15, 200...	Jan 15, 200...	Jan 15, 200...	cnt1855i.un...	0:0:0.0
24	4816	SASGrid:10173	sascnn1	Done	Jan 15, 200...	Jan 15, 200...	Jan 15, 200...	cnt1855b.u...	0:0:0.0
25	4817	SASGrid:10173	sascnn1	Done	Jan 15, 200...	Jan 15, 200...	Jan 15, 200...	cnt1855g.u...	0:0:0.0
26	4818	SASGrid:10173	sascnn1	Done	Jan 15, 200...	Jan 15, 200...	Jan 15, 200...	cnt1855d.u...	0:0:0.0
27	4812	SASGrid:10173	sascnn1	Done	Jan 15, 200...	Jan 15, 200...	Jan 15, 200...	cnt1855a.u...	0:0:0.0

Using Grid Manager, you can customize the view by selecting the columns of data to display and the order in which they should appear. In addition, you can filter, sort, and refresh the display of jobs.

Each grid that you define must have one computer with a grid monitoring server configured.

Maintaining the Grid

Viewing Grid Information

When you expand the Grid Manager node in the navigation tree, all of the grid monitoring servers that you have defined are listed under the name of the plug-in. To view information about a specific server, expand the server's node in the navigation tree. The information for a server is grouped into three categories in the navigation tree:

- Job Information
- Host Information
- Queue Information

Select a category to display a table that contains information for the category. You can also display a graph of the job information. Right-click a category in the navigation tree and select **`Properties`** from the pop-up menu to choose the columns that are displayed in the table and to choose how to filter the information that is displayed. You can also manage jobs, hosts, and queues from the tables.

Managing Jobs

Use the Grid Manager to terminate, suspend, and resume jobs.

To terminate a job, follow these steps:

1. In the selection tree, select the **`Job Information`** node.
2. In the table, locate the job that you want to cancel.
3. Right-click any column in the row for the job and select **`Terminate Task`** from the pop-up menu.

If you log on to SAS Management Console using a user ID that is defined as an LSF Administrator ID, you can terminate jobs that have been submitted to the LSF servers. Users can terminate only their own jobs. The LSF Administrator can terminate any job. If you are terminating a job on Windows, be sure to match the domain name exactly (including case).

To suspend a job (pause the job's execution), follow these steps:

1. In the selection tree, select the **Job Information** node.
2. In the table, locate the job that you want to suspend.
3. Right-click any column in the row for the job and select **Suspend Job** from the context menu.

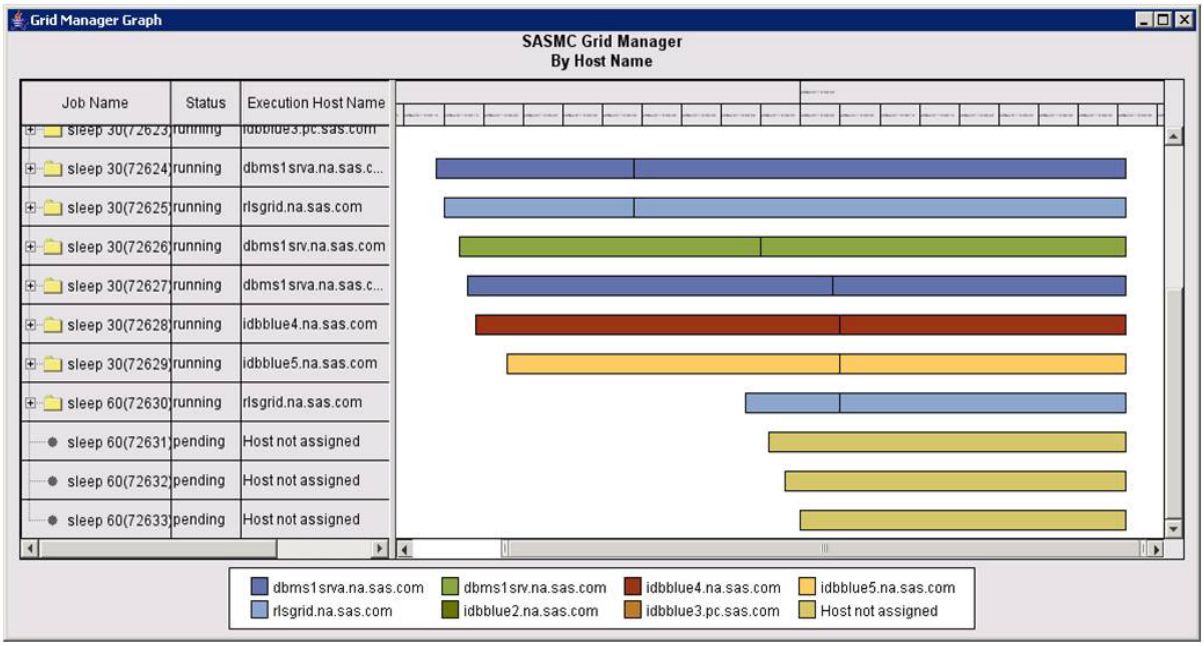
To resume processing of a suspended job, follow these steps:

1. In the selection tree, select the **Job Information** node.
2. In the table, locate the job that you want to resume.
3. Right-click any column in the row for the job and select **Resume Job** from the context menu.

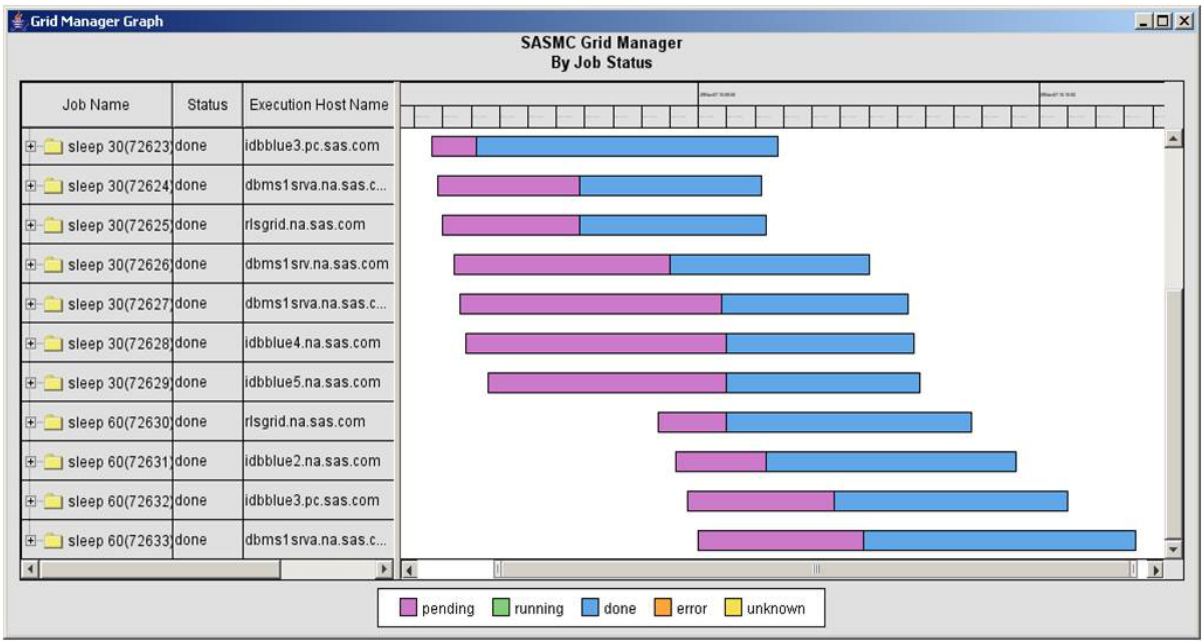
Displaying Job Graphs

You can use the Grid Manager to display GANTT charts for jobs running on the grid. To display a chart, follow these steps:

1. In the selection tree, select the Job Information node.
2. Right-click and select either **Create Graph by Host** or **Create Graph by Status** from the **Actions** menu, the context menu, or the toolbar.
3. If you select **Create Graph by Host**, a Gantt chart is displayed that shows the amount of time taken to process each job and identifies the machine on which the job ran.



4. If you select **Create Graph by Status**, a Gantt chart is displayed that illustrates the amount of time that each submitted job spent in each job status (such as pending or running).



Closing and Reopening Hosts

You can use the Grid Manager to close or reopen hosts on the grid. A closed host cannot process any jobs that are sent to the grid. Closing a host is useful when you want to remove the host from the grid for maintenance. You can also close the grid control server to prevent it from receiving work.

Note: The status of a host does not change right away after it has been opened or closed. By default, the host status is polled every 60 seconds by the Grid Management Service. The polling time interval is specified by the GA_HOST_POLL_TIME property in the ga.conf file, which is located in the **`<LSF_install_dir>/gms/conf`** directory

To close a host, follow these steps:

1. In the navigation area, open the node for the grid containing the host.
2. Select the **Host Information** node.

   The display area contains a table of the hosts in the grid.
3. In the table, right-click the host that you want to close and select **Close** from the context menu.

The host now cannot accept jobs that are sent to the grid.

To open a host that has been closed, follow these steps:

1. In the navigation area, open the node for the grid containing the host.
2. Select the **Host Information** node. The display area contains a table of the hosts in the grid.
3. In the table, right-click the host that you want to open and select **Open** from the context menu. The host can now accept jobs that are sent to the grid.

Managing Queues

You can use the Grid Manager to close, open, activate, and inactivate queues. A closed queue cannot accept any jobs that are sent to the grid. An inactive queue can still accept jobs, but none of the jobs in the queue can be processed. Closing a queue is useful when you need to make configuration changes to the queue.

To close a queue, follow these steps:

1. In the navigation area, open the node for the grid containing the queue.
2. Select the **Queue Information** node.

   The display area contains a table of the queues in the grid.
3. In the table, right-click the queue that you want to close and select **Close** from the context menu.

   The queue is now prevented from accepting jobs that are sent to the grid.

To open a closed queue, follow these steps:

1. In the navigation area, open the node for the grid containing the queue.
2. Select the **Queue Information** node.

   The display area contains a table of the queues in the grid.
3. In the table, right-click the queue that you want to open and select **Open** from the context menu.

   The queue can now accept jobs that are sent to the grid.

To inactivate a queue, follow these steps:

1. In the navigation area, open the node for the grid containing the queue.
2. Select the **Queue Information** node.

   The display area contains a table of the queues in the grid.
3. In the table, right-click the active queue that you want to make inactive and select **Inactivate** from the context menu.

To activate a queue, follow these steps:

1. In the navigation area, open the node for the grid containing the queue.
2. Select the **Queue Information** node.

   The display area contains a table of the queues in the grid.
3. In the table, right-click the inactive queue that you want to make active and select **Activate** from the context menu.

Using Platform RTM for SAS

Platform RTM for SAS is a Web-based tool that lets you graphically view the status of devices and services within your SAS Grid environment as well as manage the policies and configuration of the grid. It is a visual tool to quickly track and diagnose issues before they affect service levels. Platform RTM for SAS provides these features:

- drill-down capabilities to view details of hosts, jobs, queues, and user activities
- instant alerts on job performance and grid efficiency to enable administrators to optimize usage and workloads
- customizable graphs to visually analyze resource usage, workload trends, and job behavior
- interfaces to allow administrators to update the policies and rules in the grid configuration as well as set up high availability for any grid services as well as the applications executing in the grid

Platform RTM for SAS helps system administrators improve decision-making, reduce costs and increase service levels for SAS grid deployments. Refer to the documentation included with the Platform RTM installation package for instructions.

Chapter 6

Troubleshooting

Overview of the Troubleshooting Process

These topics provide the framework for a systematic, top-down approach to analyzing problems with a grid environment. By starting at the highest level (the network) and working downward to the job execution, many common problems can be eliminated.

For the troubleshooting information not contained here, go to **http://support.sas.com/rnd/scalability/grid/gridinstall.html** or contact SAS Technical Support.

Verifying the Network Setup

Overview

The first step in troubleshooting problems with a SAS grid is to verify that all computers in the grid can communicate with one another through the ports that are used by the grid middleware.

Host Addresses

Check the /etc/hosts file on each grid node to ensure that the machine name is not mapped to the 127.0.0.1 address. This mapping causes the signon connection to the grid node to fail or to hang. This happens because the SAS session being invoked on the grid node cannot determine the correct IP address of the machine on which it is running. A correct IP address must be returned to the client session in order to complete the connection. For example, delete the name "myserver" if the following line is present in the /etc/hosts file

```
127.0.0.1 myserver localhost.localdomain localhost
```

Host Connectivity

You must verify that the network has been set up properly and that each machine knows the network address of all the other machines in the grid. Follow these steps to test the network setup:

1. Run the **`hostname`** command on every machine in the grid (including grid nodes, grid control servers, and Foundation SAS grid clients).
2. Run the **`ping`** command on all grid node machines and the grid control machine against every other machine in the grid (including grid client machines). When you ping a grid client machine, use the host name without the domain suffix.
3. Run the **`ping`** command on each grid client machine against every other machine in the grid (including itself). When a grid client machine pings itself using the value from the hostname command, verify that the returned IP address is the same IP address that is returned when the grid nodes ping the client. However, this might not occur on machines with multiple network adapters.

If the network tests indicate a problem, you must either correct the DNS server or add entries to each machine's hosts file. Contact your network administrator for the best way to fix the problem.

Platform LSF assumes that each host in the grid has a single name, that it can resolve the IP address from the name, and that it can resolve the official name from the IP address. If any of these conditions are not met, LSF needs its own **`hosts`** file, which is located in its configuration directory (LSF_ENVDIR/conf/hosts).

Host Ports

You must verify that the ports that SAS and LSF use for communication are accessible from other machines. The ports might not be accessible if a firewall is running on one or more machines. If firewalls are running, you must open ports so that communication works between the LSF daemons and the instances of SAS. Issue the **`telnet`** <*host*><*port*> command to determine whether a port is open on a specific host.

The default ports used in a grid are:

- LSF: 6878, 6881, 6882, 7869, 7870, 7871, 7872
- Grid Monitoring Service: 1976
- Platform Process Manager: 1966

If you need to change any port numbers, modify these files:

- LSF ports: LSF_ENVDIR/conf/lsf.conf and EGO_CONFDIR/ego.conf

- Grid Monitoring Service port: gms/conf/ga.conf
- Platform Process Manager port: pm/conf/js.conf

If you change the Grid Monitoring Service port, you must also change the metadata for the Grid Monitoring Server. If you change the Platform process Manager port, you must also change the metadata for the Job Scheduler Server.

Ports might be used by other programs. To check for ports that are in use, stop the LSF daemons and issue the command **`netstat -an |`**<*search-tool*><*port*>, where *search-tool* is grep (UNIX) or findstr (Windows). Check the output of the command for the LSF ports. If a port is in use, reassign the port or stop the program that is using the port.

SAS assigns random ports for connections, but you can restrict the range of ports SAS uses by using the **`-tcpportfirst <`*`first-port`*`>`** and the **`-tcpportlast <`*`last-port`*`>`** options. You can specify these options in the SAS configuration file or on the SAS command line. For remote sessions, you must specify these options either in the grid command script (sasgrid.cmd on Windows or sasgrid on UNIX) or in the **Command** field in the logical grid server definition in metadata. For example, adding the following parameters to the SAS command line in the grid script restricts the ports that the remote session uses to between 5000 and 5005:

```
-tcpportfirst 5000 -tcpportlast 5005
```

Verifying the Platform Suite for SAS Environment

Verifying That LSF Is Running

After the installation and configuration process is complete, verify that all of the LSF daemons are running on each machine.

For Windows machines, log on to each machine in the grid and check the Services dialog box to verify that these services are running:

- Platform LIM
- Platform RES
- Platform SBD

For UNIX machines, log on to each machine in the grid and execute the **`ps`** command to check for processes that are running in a subdirectory of the $LSF_install_dir. An example command is:

```
ps -ef|grep LSF_install_dir
```

The daemons create log files that can help you to debug problems. The log files are located in the machine's *LSF_install_dir*\logs directory (Windows) or the shared LSF_TOP/log directory (UNIX). If the daemon does not have access to the share on UNIX, the log files are located in the /tmp directory.

If the command fails, check the following:

- Verify that the path to the LSF programs is in the PATH environment variable. For LSF 7, the path is *LSF_install_dir*/7.0/bin.
- On UNIX machines, you might have to source the **`LSF_TOP/conf/profile.lsf`** file to set up the LSF environment.

- A machine might not be able to access the configuration files. Verify that the machine has access to the shared directory that contains the binary and configuration files, defined by the LSF_ENVDIR environment variable. If the file server that is sharing the drive starts after the grid machine that is trying to access the shared drive, the daemons on the machine might not start. Add the LSF_GETCONF_TIMES environment variable to the system environment and set the variable value to the number of times that you want the daemon to try accessing the share in each five-second interval before the daemon quits. For example, setting the variable to a value of 600 results in the node trying for 50 minutes ((600*5 seconds)/60 seconds per minute) before quitting.
- The license file might be invalid or missing. If LSF cannot find a license file, some daemons might not start or work correctly. Make sure that the license file exists, is properly referenced by the LSF_LICENSE_FILE parameter in the LSF_ENVDIR/conf/lsf.conf file, and is accessible by the daemons.
- All daemons might not be running. Restart the daemons on every machine in the grid using the **`lsfrestart`** command. If this command does not work, run the /etc/init.d/lsf restart command (UNIX) or use the Services Administration tool (Windows). Open Services Administration, stop the SBD, RES, and LIM services (in that order). Next, start the LIM, RES, and SBD services (in that order).
- A grid machine might not be able to connect to the SAS grid control machine. The grid control machine is the first machine listed in the lsf.cluster.<*cluster_name*> file. Make sure that the daemons are running on the master host and verify that the machines can communicate with each other.

Verifying LSF Setup

You must verify that all grid machine names are specified correctly in the LSF_ENVDIR/conf/lsf.cluster.<*cluster_name*> file and the resource is specified in the lsf.shared file. Follow these steps to make sure the configuration is correct:

1. Log in as an LSF administrator on one of the machines in the grid, preferably the grid control server machine. The LSF administrator ID is listed in the lsf.cluster.<*cluster_name*> file under the line **`Administrators=`*`username1username2 ... usernameN`***.
2. Run the command **`lsadmin ckconfig -v`** to check the LSF configuration files for errors.
3. Run the command **`badmin ckconfig -v`** to check the batch configuration files for errors.
4. Run the command **`lshosts`** to list all the hosts in LSF and to verify that all the hosts are listed with the proper resources.
5. Run the command **`bhosts`** to list all the hosts in LSF's batch system. Verify that all hosts are listed. Make sure that the Status for all hosts is set to **`ok`** and that the MAX column has the correct number of jobs slots defined for each host (the maximum number of jobs the host can process at the same time).
6. If you find any problems, correct the LSF configuration file and issue the commands **`lsadmin reconfig`** and **`badmin reconfig`** so that the daemons use the updated configuration files.
7. If you added or removed hosts from the grid, restart the master batch daemon by issuing the command **`badmin mbdrestart`**. To restart everything, issue the **`lsfrestart`** command.

Verifying LSF Job Execution

Some problems occur only when you run jobs on the grid. To minimize and isolate these problems, you can run debug jobs on specific machines in the grid.

To submit the debug job, run the command **bsub -I -m <*host_name*> set** from the grid client machine to each grid node. This command displays the environment for a job running on the remote machine and enables you to verify that a job runs on the machine.

If this job fails, run the **bhist -l <*job_id*>**' command, where *job_id* is the ID of the test job. The output of the command includes the user name of the person submitting the job, the submitted command, and all the problems LSF encountered when executing the job. Some messages in the bhist output for common problems are:

Failed to logon user with password
: specifies that the password in the Windows **passwd.lsfuser** file is invalid. Update the password using the **lspasswd** command.

Unable to determine user account for execution
: specifies that the user does not have an account on the destination machine. This condition can occur between a Windows grid client to a UNIX grid node, because the Windows user has a domain prefixed to the user name. Correct this problem by making sure that the user has an account on the UNIX machines. Also, add the line **LSF_USER_DOMAIN=** to the Windows lsf.conf file to strip the domain from the user name.

Verifying the SAS Environment

Verifying SAS Grid Metadata

SAS needs to retrieve metadata about the grid from a SAS Metadata Server in order to operate properly. Start the SAS Management Console and use the Server Manager plug-in to verify the following:

Logical grid server
: Under the SAS Application Server context (for example, SASApp), verify that a logical grid server has been defined.

  Open the Properties window for the logical grid server and verify that the properties contain the correct path to the script file or the correct command that is executed on the grid node. Verify that the path exists on every node in the grid and that the command is valid on every node in the grid.

Grid monitoring server
: Verify that a grid monitoring server has been defined.

  Open the connection properties for the server and verify that the properties contain the name or address of the machine that is running the Grid Monitoring Server daemon (typically the SAS grid control machine). Verify that the port specified in the properties is the same as that specified in the Grid Monitoring Service configuration file (the default value is 1976).

Verifying Grid Monitoring

The Grid Manager plug-in for SAS Management Console displays information about the grid's jobs, hosts, and queues. After you define the Grid Monitoring Server and the Grid Management Service is running on the control server, grid information is displayed in the Grid Manager plug-in in SAS Management Console. Common error messages encountered in the Grid Manager plug-in include:

Connection timed out or Connection refused
: The Grid Management Service is not running. Start the Grid Management Service on the grid control machine.

Your userid or password is invalid. Please try again or contact your systems administrator
: The user provided invalid credentials for the machine running the Grid Monitoring Service or the user's credentials that are stored in the metadata do not include a password for the login associated with the authorization domain used by the Grid Monitoring Server connection. For example, "Grid 1 Monitoring Server" is defined in the metadata to use the "DefaultAuth" authorization domain. The user "User1" has a login defined in the User Manager for the "DefaultAuth" domain, but the login has only the user ID specified and the password is blank.

  To correct the problem, either provide complete credentials for the authorization domain for the user, remove the login for the authorization domain, or use a different authorization domain for the grid monitoring server connection. If you provide the correct credentials, the user is not prompted for a user ID and password. If you remove the login for that authorization domain or change the grid monitoring server connection to use a different authorization domain without adding credentials for the user for that domain, the user is prompted for their user ID and password to connect to the machine where the grid monitoring server is running.

Verifying SAS Job Execution

SAS provides a grid test program on the SAS support Web site tests connectivity to all nodes in the grid. Run the program from a grid client. You can download the program from **http://support.sas.com/rnd/scalability/grid/gridfunc.html#testprog**. After you download the program, follow these steps:

1. Copy and paste the grid test program into a Foundation SAS Display Manager Session.
2. If the application server associated with your logical grid server in your metadata is not named “SASMain”, change all occurrences of “SASMain” in the test program to the name of the application server associated with your logical grid server. For example, some SAS installations have the application server named “SASApp”, so all occurrences of SASMain should be replaced with SASApp.
3. Submit the code.

The program attempts to start one remote SAS session for every job slot available in the grid. The program might start more than one job on multi-processor machines, because LSF assigns one job slot for each core by default.

Here are some problems you might encounter when running the grid test program:

`Grid Manager not licensed` message
: Make sure that your SID contains a license for SAS Grid Manager.

`Grid Manager cannot be loaded` message
Make sure that Platform Suite for SAS has been installed and the LSF and PATH environment variables are defined properly.

`Invalid resource requested` message
The application server name or workload value has not been defined in the lsf.shared file. Also make sure you associate the value with the hosts you want to run SAS programs in the lsf.cluster.<*cluster_name*> file.

The number of grid nodes is 0.
Possible reasons for this error include:

- The application server name was not defined as a resource name in the lsf.shared file.
- The application server name was not associated with any grid nodes in the lsf.cluster<*cluster_name*>. file.
- The grid client where the job was submitted cannot communicate with the entire grid.

The number of grid nodes is not the same as the number of grid node machines.
As shipped, the number of grid nodes equals the number of job slots in the grid. By default, the number of job slots is equal to the number of cores, but the number of job slots for a grid node can be changed.

Another explanation is that the application server name has not been associated with all the grid nodes in the lsf.cluster.<*cluster_name*> file.

Jobs fail to start.
Possible reasons for this problem include:

- The grid command defined in the logical grid server metadata is either not valid on grid nodes or does not bring up SAS on the grid node when the command is run. To verify the command, log on to a grid node and run the command defined in the logical grid server definition. The command should attempt to start a SAS session on the grid node. However, the SAS session does not run successfully, because grid parameters have not been included. Platform Suite for SAS provides a return code of 127 if the command to be executed is not found and a return code of 128 return code if the command is found, but there is a problem executing the command.
- Incorrect version of SAS installed on grid nodes. SAS 9.1.3 Service Pack 3 is the minimum supported version. A return code of 231 might be associated with this problem.
- Unable to communicate between the grid client and grid nodes. Verify that the network is set up properly, using the information in "Verifying the Network Setup " on page 57.

Jobs run on machines that are supposed to be only grid clients.
By default, all machines that are listed in the lsf.cluster.<*cluster_name*> file are part of the grid and can process jobs. If you want a machine to be able to submit jobs to the grid (a grid client) but not be a machine that can process the job (a grid node), set its maximum job slots to 0 or use the Grid Manager plug-in to close the host.

Part 2

SAS Grid Language Reference

Chapter 7

SAS Functions for SAS Grid

Dictionary

GRDSVC_ENABLE Function

Enables or disables one or all SAS sessions on a grid.

Valid in: %SYSFUNC or %QSYSFUNC Macro, DATA step

Category: Grid

Syntax

grdsvc_enable(*identifier* <,*option-1*; ... *option-n*>)

grdsvc_enable(*identifier*,"" | '')

Required Argument

identifier

specifies one or all server sessions to be enabled or disabled for grid execution. The *identifier* is specified as follows:

server-ID

specifies the name of a SAS/CONNECT server session to be enabled or disabled for grid execution.

You use this *server-ID* when you sign on to a server session using the SIGNON or the RSUBMIT statement. For information about ways to specify the server ID, see *SAS/CONNECT User's Guide*.

Requirement: If the function is used in a DATA step, enclose *server-ID* in double or single quotation marks. A *server-ID* cannot exceed eight characters.

ALL
specifies that all SAS sessions are enabled or disabled for grid execution.

See: SIGNON statement and RSUBMIT statement in *SAS/CONNECT User's Guide*

Example:

```
%let rc=%sysfunc(grdsvc_enable(grdnode1,server=SASApp));
   %let rc=%sysfunc(grdsvc_enable(_all_,server=SASApp));
   %let rc=%sysfunc(grdsvc_enable(notgrid1,""));
```

Optional Arguments

SASAPPSERVER=*server-value*
specifies the name of a SAS Application Server that has been defined in the SAS Metadata Repository. The SAS Application Server contains the definition for the logical grid server that defines the grid environment.

Alias: SERVER=, RESOURCE=

Restriction: Although a SAS Application Server is configured as a required grid resource in most environments, some grids are not partitioned by resource names. In these environments, passing the SAS Application Server name as a required resource causes the job to fail. To find out whether the SAS Application Server is designated as a required resource value or not in the SAS Metadata Repository, use the GRDSVC_GETINFO function call.

Interaction: The name of the SAS Application Server is passed to the grid middleware as a resource value. When the job is executed, the grid middleware selects a grid node that meets the requirements that are specified by this value. If *SAS-application-server* contains one or more spaces, the spaces are converted to underscores before the name is passed to the grid middleware as a resource value.

Tip: For Platform Suite for SAS, this *server-value* corresponds with the value of a resource that the LSF administrator has configured in the lsf.cluster.*cluster-name* file and the lsf.shared file on the grid-control server.

See: "GRDSVC_GETINFO Function" on page 72 to find out whether the SAS Application Server is designated as a required resource value in the SAS Metadata Repository.To remove the SAS Application Server name as a required resource, see "Modifying SAS Logical Grid Server Definitions" on page 17.

Example:

```
%let
   rc=%sysfunc(grdsvc_enable(_all_, server=SASApp));
```

WORKLOAD=*workload-value*
identifies the resource for the job to be executed on the grid. This value specifies an additional resource requirement for which the grid middleware selects the appropriate grid nodes.

The specified workload value should match one of the workload values that are defined in the SAS Application Server in the SAS Metadata Repository.

Requirement: Workload values are case sensitive

Interaction: If *workload-value* contains one or more spaces, the spaces are converted to underscores before the value is passed to the grid provider. If *workload-value* is not located in the SAS Application Server definition and no other errors occur, a 0 result code is returned, and this note is displayed:

```
NOTE: Workload value "gridResource" does not exist in the SAS Metadata Repository
```

.

Tip: For Platform Suite for SAS, this *workload-value* corresponds with the resource that the LSF administrator has configured in the lsf.cluster.*cluster-name* file and the lsf.shared file on the grid-control computer.

Example:

```
%let
   rc=%sysfunc(grdsvc_enable(grdnode1, server=SASApp;
   workload=EM));
```

The workload value EM specifies the resource name. EM must be assigned to a grid node in order to process this job. An example is assigning EM to machines that can process SAS Enterprise Miner jobs.

JOBNAME=*job-name-macro-variable*
specifies the macro variable that contains the name that is assigned to the job that is executed on the grid.

Example:

```
%let
   hrjob=MyJobName;
   %let rc=%sysfunc(grdsvc_enable(grdnode1, server=SASApp;
   jobname=hrjob));
    signon grdnode1;
```

In this example, hrjob is the name of the macro variable to which the job name is assigned. The actual job name is MyJobName. The status of the job can be tracked using the SAS Grid Manager Plug-in for SAS Management Console. In this example, you track the status of the job named MyJobName.

JOBOPTS=*job-opts-macro-variable*
specifies the macro variable that contains the job options. The job option name/value pairs are assigned to *job-opts-macro-variable*.

The job options are used by the grid job to control when and where a job runs. Job options vary according to the grid middleware provider. Job options are specified as name/value pairs in this format:

```
option-1=value-1;option-2="value-2 with spaces"; ...
option-n='value-n with spaces';
```

For a list of the job options you can specify, see "Supported Job Options " on page 89.

Requirement: Use a semicolon to separate job option/value pairs. For multiple values, use a macro quoting function for the semicolon or use single or double quotation marks to enclose all job options. If the value contains one or more spaces, tabs, semicolons, or quotation marks, enclose the value in single or double quotation marks

See: For job options that are provided by middleware providers other than Platform Computing, such as Data Synapse and Univa UD, see http://support.sas.com/rnd/scalability/grid. For details about using the quoting macro function, see *SAS Macro Language: Reference*.

Example:

```
%let
   rc=%sysfunc(grdsvc_enable(all, server=SASApp; jobopts=hrqueue));
   %let hrqueue=queue=priority%str(;)project="HR Monthly";
   signon grdnode1;
   %let hrqueue='queue=priority;project="HR Yearly"';
   signon grdnode2
```

Both jobs are sent to the priority queue. The first job is associated with the project named “HR Monthly” and the second job is associated with the project named “HR Yearly.”

"" | ''
disables grid execution for the specified server ID or all server sessions.

This value is intended to be used when you have specified _ALL_ in a previous call but you want to disable it for a small number of exceptions.

Requirement: Double or single quotation marks can be used. Do not insert a space between the double or single quotation marks.

Interaction: When quotation marks are used with _ALL_, it clears all previous grid settings that were specified using the GRDSVC_ENABLE function.

Example:

```
%let rc=%sysfunc(grdsvc_enable(grdnode1,""));
   %let rc=%sysfunc(grdsvc_enable(_all_,''));
```

Details

The GRDSVC_ENABLE function is used to enable and disable a grid execution. Grid execution can be enabled for a specified SAS session or for all SAS grid sessions. If a grid environment is not configured or is unavailable, the job is started as a symmetric multi-processor (SMP) process instead.

The GRDSVC_ENABLE function does not resolve to a specific grid node, and it does not cause grid execution. The server ID is mapped to a specific grid node. The server session starts on the grid node when requested by subsequent SAS statements (for example, when the SIGNON statement or the RSUBMIT statement is executed).

In order to restrict the use of specific grid nodes to be used by server sessions, the name of the SAS Application Server and the workload resource value are passed as required resources to the grid middleware.

Note: An exception to this behavior is when the SAS Application Server is disabled as a required resource for the grid server. For details, see the restriction for the SASAPPSERVER= option.

The grid can be partitioned according to resource or security requirements. If grid nodes do not have the required resources, then SAS requests fail. If grid nodes have the required resources but are busy, SAS requests are queued until grid resources become available. For information, see “Partitioning the Grid ” on page 28.

Some SAS applications are suited for execution in a grid environment, but not in an SMP environment. Such applications should contain a macro that checks the return code from the GRDSVC_ENABLE function to ensure that a grid node, rather than an SMP process, is used.

Here are the result codes:

Table 7.1 *GRDSVC_ENABLE Function Result Codes*

Result Code	Explanation
2	Reports that one or all server sessions were disabled from grid execution.

Result Code	Explanation
1	Reports that a grid environment is unavailable due to one or more of these conditions: • A connection to the SAS Metadata Server is unavailable. • A logical grid server has not been defined in the SAS Metadata Repository. • The current user identity does not have authorization to use the specified logical grid server. • SAS Grid Manager has not been licensed. Instead, server sessions execute on the multi-processor (SMP) computers as a SASCMD sign-on. One of these commands, in order of precedence, is used to start the server session: • the value of the SASCMD system option • `!sascmd -noobjectserver`
0	Reports that the specified session was enabled.
-1	Reports a syntax error in the function call. An example is the omission of the server ID.
-2	Reports a parsing error in the function call. An example is an invalid option.
-3	Reports an invalid server ID in the function call.
-5	Reports an out-of-memory condition while the function is executing.

See Also

- *SAS/CONNECT User's Guide*
- *SAS/CONNECT User's Guide*
- *SAS Language Reference: Dictionary*
- *SAS Macro Language: Reference*

GRDSVC_GETADDR Function

Reports the IP address of the grid node on which the SAS session was chosen to execute.

Valid in: %SYSFUNC or %QSYSFUNC Macro, DATA step

Category: Grid

Syntax

grdsvc_getaddr(*identifier*)

Without Arguments

Required Argument

identifier

identifies the server session that is executing on the grid. The identifier can be specified as follows:

""| "
is an empty string that is used to refer to the computer on which the function is executing.

server-ID
specifies the server session that is executing on a grid.

You use the same *server-ID* that was used to sign on to a server session using the RSUBMIT statement or the SIGNON statement. Each server ID is associated with a fully qualified domain name (FQDN). The name resolution system that is part of the TCP/IP protocol is responsible for associating the IP address with the FQDN. The output is one or more IP addresses that are associated with the server. IP addresses are represented in IPv4 and IPv6 format, as appropriate.

Requirement: Double or single quotation marks can be used. Do not insert a space between the double or single quotation marks.

Interaction: If the function is used in a DATA step, enclose *server-ID* in double or single quotation marks.

Example

```
/*-------------------------------------------------------------------*/
/* The following sets the macro variable 'myip' to the IP address    */
/* of the grid node associated with the server session 'task1'        */
/*------------------------------------------------------------------- */
  %let
myip=%sysfunc(grdsvc_getaddr(task1));
```

See Also

RSUBMIT statement

- *SAS/CONNECT User's Guide*

SIGNON statement

- *SAS/CONNECT User's Guide*

DATA step

- *SAS Language Reference: Dictionary*

%SYSFUNC or %QSYSFUNC

- *SAS Macro Language: Reference*

GRDSVC_GETINFO Function

Reports information about the grid environment.

Valid in: %SYSFUNC or %QSYSFUNC Macro, DATA step

Category: Grid

Syntax

grdsvc_getinfo(*identifier*)

Required Argument

identifier

specifies the server session or the SAS Application Server whose details you want to have reported to the SAS log.

The *identifier* is specified as follows:

server-ID
reports details about the specified server ID. The details that are returned by the GRDSRV_INFO function reflect the arguments that are specified in the GRDSVC_ENABLE function. You can request details about a **`server-ID`** that you have used to create a server session or that you will use to create a server session on the grid.

Requirement: A *server-ID* cannot exceed eight characters.

ALL
reports details about all server IDs to the SAS log. The details that are returned by the GRDSRV_INFO function reflect the arguments that are specified in the GRDSVC_ENABLE function.

SASAPPSERVER=*SAS-application-server*
reports information about the specified SAS Application Server to the SAS log.

Alias: SERVER=, RESOURCE=

SHOWID
lists each server session and its status: enabled for grid execution, enabled for SMP execution, or disabled.

Interaction: If the GRDSVC_GETINFO function is used in a DATA step, enclose the *identifier* in single or double quotation marks. The identifier can be specified as *server-ID*, _ALL_, SASAPPSERVER=*SAS-application-server*, or _SHOWID_. If no grid processes were enabled using the GRDSRV_ENABLE function or if all grid processes were disabled using the GRDSVC_ENABLE function with _ALL_ option, this message is displayed:

```
NOTE: No remote session ID enabled/disabled for the grid service.
```

Tip: You do not have to be signed on to a specific server session in order to get information about it.

Example: This log message reports that the SAS Application Server is a required resource.

```
%put
   %sysfunc(grdsvc_getinfo(server=SASApp));
   NOTE: SAS Application Server Name= SASAPP
               Grid Provider=   Platform
               Grid Workload=   gridwrk
               Grid SAS Command=   gridsasgrid
               Grid Options=   gridopts
               Grid Server Addr=   d15003.na.sas.com
               Grid Server Port=   123
```

```
Grid Module=   gridmod
Server name is a required grid resource value.
```

If the SAS Application Server is a disabled required resource, this message is displayed:

```
Server name is not a required grid resource value.
```

Details

Here are the result codes:

Table 7.2 *GRDSVC_GETINFO Function Return Codes*

Result Code	Explanation
2	Reports that the specified server ID is not enabled for grid execution.
1	Reports that the specified server ID is enabled for SMP execution.
0	Reports that the specified server ID is enabled for a grid execution or that no error occurred.
-1	reports a syntax error in the function call. An example is that an empty string is specified for the server ID.
-2	Reports a parsing error in the function call. An example is the failure to specify the SAS Application Server using the SASAPPSERVER= option.
-3	Reports an invalid server ID in the function call.
-5	Reports an out-of-memory condition while the function is executing.
-6	Reports that an error occurred when the SAS Metadata Server was accessed or when the information was returned from the SAS Metadata Server

Example

```
/*-------------------------------------------------------------------------*/
/* Show grid logical server definition for SAS Application Server 'SASApp'*/
/*-------------------------------------------------------------------------*/
%let rc=%sysfunc(grdsvc_getinfo(sasappserver=SASApp));
/*-------------------------------------------------------------------------*/
/* Show grid information about server session ID 'task1'                   */
/*-------------------------------------------------------------------------*/
%let rc=%sysfunc(grdsvc_getinfo(task1));
/*-------------------------------------------------------------------------*/
/* Show server session information for all server sessions                 */
/*-------------------------------------------------------------------------*/
%let rc=%sysfunc(grdsvc_getinfo(_ALL_));
/*-------------------------------------------------------------------------*/
/* Show all server session IDs that are either grid-enabled or             */
/* grid-disabled                                                           */
/*-------------------------------------------------------------------------*/
```

```
%let
rc=%sysfunc(grdsvc_getinfo(_SHOWID_));
```

See Also

RSUBMIT statement

- *SAS/CONNECT User's Guide*

SIGNON statement

- *SAS/CONNECT User's Guide*

DATA step

- *SAS Language Reference: Dictionary*

%SYSFUNC or %QSYSFUNC

- *SAS Macro Language: Reference*

GRDSVC_GETNAME Function

Reports the name of the grid node on which the SAS grid server session was chosen to execute.

Valid in: %SYSFUNC or %QSYSFUNC Macro, DATA step

Category: Grid

Syntax

grdsvc_getname(*identifier*)

Required Argument

identifier

identifies the server session that is executing on the grid. The identifier can be specified as follows:

"" | ''
is an empty string that is used to refer to the computer at which the statement is executed.

server-ID
specifies the server session that is executing on a grid.

You use the same *server-ID* that you used to sign on to a server session using the RSUBMIT statement or the SIGNON statement .

If the function is used in a DATA step, enclose *server-ID* in double or single quotation marks.

Example

```
/*-------------------------------------------------------------------------*/
/* The following sets the macro variable 'mynodea' to the name of        */
/* the grid node associated with the server ID 'task1'.                   */
/*-------------------------------------------------------------------------*/
%let
mynodea=%sysfunc(grdsvc_getname(task1));
```

See Also

RSUBMIT statement

- *SAS/CONNECT User's Guide*

SIGNON statement

- *SAS/CONNECT User's Guide*

DATA step

- *SAS Language Reference: Dictionary*

%SYSFUNC or %QSYSFUNC

- *SAS Macro Language: Reference*

GRDSVC_NNODES Function

Reports the total number of job slots that are available for use on a grid.

Valid in: %SYSFUNC or %QSYSFUNC Macro, DATA step

Category: Grid

Syntax

grdsvc_nnodes(*argument*;*option*)

Without Arguments

Required Argument

SASAPPSERVER=*SAS-application-server*

specifies the name of the SAS Application Server that has been defined in the SAS Metadata Repository. The SAS Application Server contains the definition for the logical grid server that is used to access the grid environment. The name of the SAS Application Server is passed to the grid middleware as a required resource. The grid middleware selects the grid nodes that meet the requirements for the specified SAS Application Server and returns the total number of job slots in the grid.

An exception to this behavior is when the SAS Application Server is disabled as a required resource for the grid server. For details see the SASAPPSERVER= option for the GRDSVC_ENABLE function on page 67.

Alias: SERVER=, RESOURCE=

Interaction: If *SAS-application-server* contains one or more spaces, the spaces are converted to underscores before the name is passed to the grid middleware.

Example:

```
%let
   numofnodes%sysfunc(grdsvc_nnodes(server=SASApp));
```

Optional Argument

WORKLOAD=*workload-value*

identifies the resource for the type of job to be executed on the grid. This value specifies the workload requirements for which the grid middleware selects the grid nodes that contain these resources.

The specified workload value should match one of the workload values that is defined in the SAS Application Server in the SAS Metadata Repository.

Requirement: If you specify WORKLOAD=, you must also specify the SASAPPSERVER= option. Workload values are case sensitive.

Interaction: If *workload-value* contains one or more spaces, the spaces are converted to underscores before the value is passed to the grid middleware. If *workload-value* is not located in the SAS Application Server definition and no other errors occur, a 0 result code is returned. A 0 result code means that no grid nodes contain the requested resources. Also, this note is displayed:

```
NOTE: Workload value "gridResource" does not exist in the SAS Metadata Repository.
```

If *workload-value* is undefined to the grid middleware, the GRDSVC_NNODES function returns the result code 0.

Tip: For Platform Suite for SAS, this *workload-value* corresponds with the resource that the LSF administrator has configured in the lsf.cluster.*cluster-name* file and the lsf.shared file on the grid-control computer.

Example:

```
%let
   numofnodes=%sysfunc(grdsvc_nnodes(server=SASApp; workload=em));
```

The workload value, EM , specifies the resource name. EM must be assigned to a grid node in order to process this job. An example is assigning EM to machines that can process SAS Enterprise Miner jobs.

Details

When a grid environment is available, the GRDSVC_NNODES function returns the total number of job slots (busy and idle) that are available for job execution. This value is resolved at the time that the function is called. Because of this, the value might vary over time, according to whether job slots have been added or removed from the grid.

Here are the result codes:

Table 7.3 *GRDSVC_NNODES Function Result Codes*

Result Code	Explanation
nnn	If a grid environment is available, reports the total number of job slots (idle and busy) that have been configured in a grid environment. The grid contains the resources that are specified by the SASAPPSERVER= argument and the WORKLOAD= option. If a grid environment is not available, assumes a multi-processor (SMP) environment, and reports the value of the CPUCOUNT system option. In this case, the lowest value that can be reported is 1.
1	If a grid environment is not available, assumes a multi-processor (SMP) environment, and reports the value of the CPUCOUNT system option. In this case, the lowest value that can be reported is 1.
0	reports that no grid nodes contain the requested resources.
-1	reports a syntax error in the function call. For example, a syntax error would result from supplying no value, or an empty string, to the SASAPPSERVER= option.

Example

```
/*------------------------------------------------------------------------*/
/* Get the number of grid nodes that have 'SASApp' as a resource         */
/*------------------------------------------------------------------------*/
%let NumNodes=%sysfunc(grdsvc_nnodes(server=SASApp));
/*------------------------------------------------------------------------*/
/* Get the number of grid nodes that have 'SASApp'  'EM' as resources    */
/*------------------------------------------------------------------------*/
%let
numofnodes=%sysfunc(grdsvc_nnodes(server=SASApp;workload=EM));
```

See Also

RSUBMIT statement

- *SAS/CONNECT User's Guide*

SIGNON statement

- *SAS/CONNECT User's Guide*

DATA step

- *SAS Language Reference: Dictionary*

CPUCOUNT= system option

- *SAS Language Reference: Dictionary*

Chapter 8

SASGSUB Command

SASGSUB Overview

SAS Grid Manager Client Utility is a command-line utility that enables users to submit SAS programs to a grid for processing. This utility allows a grid client to submit SAS programs to a grid without having SAS installed on the machine performing the submission. It also enables jobs to be processed on the grid without requiring that the client remain active.

You can use the SAS Grid Manager Client Utility's SASGSUB command to submit jobs to the grid, view job status, retrieve results, and terminate jobs. The SAS Grid Manager Client Utility options can be specified in a configuration file so that they do not have to be entered manually. By default, SASGSUB looks for a configuration file named sasgsub.cfg in the current directory. The SAS Deployment Wizard automatically creates a configuration file that includes most of the required options. It stores the file in **<*config_dir*>/Applications/SASGridManagerClientUtiliy/<*version*>**. This is the location where you should run the SASGSUB command.

Dictionary

SASGSUB Syntax: Submitting a Job

The following is the complete syntax for submitting a SAS program to a grid. Enter the command on a Windows or UNIX command line.

Syntax

SASGSUB

-GRIDAPPSERVER *sas-application-server*
-GRIDLICENSEFILE *grid-enabled-license-file*
-GRIDSUBMITPGM *sas-program-file* -GRIDWORK *work-directory*
-JREOPTIONS *java-runtime-options* -METASERVER *server*
-METAUSER *user-ID* -METAPORT*port*
-METAPASS *password* -METAPROFILE*profile-name*
-METACONNECT *connection-name*
<-GRIDCONFIG *grid-option-file*>
<-GRIDFILESIN *grid-file-list*> <-GRIDJOBNAME *grid-program-name*>
<-GRIDJOBOPTS *grid-provider-options*>
<-GRIDPASSWORD *grid-logon-password*>
<-GRIDPLUGINPATH *grid-jar-file-path*> <-GRIDRESTARTOK>
<-GRIDSASOPTS *grid-sas-options*> <-GRIDUSER *grid-logon-username*>
<-GRIDWORKLOAD *grid-resource-names*>
<-GRIDWORKREM *shared-file-system-path*>
<-LOGCONFIGLOC *logging-option-file*><-GRIDLIBPATH *path*> <-VERBOSE>

Required Arguments

-GRIDAPPSERVER *sas-application-server*
specifies the name of the SAS Application Server that contains the grid's logical grid server definition. This option is stored in the configuration file that is automatically created by the SAS Deployment Wizard.

-GRIDLICENSEFILE *grid-enabled-license-file*
specifies the path and filename of a SAS license file that contains the SAS Grid Manager license. The default value is "license.sasgsub" and the default location is the GRIDWORK directory. You must copy the license file for the grid control server to the GRIDWORK directory and rename the file license.sasgsub in order to match the default values for this option. This option is stored in the configuration file that is automatically created by the SAS Deployment Wizard.

-GRIDSUBMITPGM *sas-program-file*
specifies the path and filename of the SAS program that you want to run on the grid.

-GRIDWORK *work-directory*
specifies the path for the shared directory that the job uses to store the program, output, and job information. The path cannot contain spaces. This option is stored in the configuration file that is automatically created by the SAS Deployment Wizard.

-JREOPTIONS *java-runtime-options*
specifies any Java run-time options that are passed to the Java Virtual Machine. This argument is required if you are using a grid provider other than Platform Suite for SAS. This option is stored in the configuration file that is automatically created by the SAS Deployment Wizard.

-METASERVER *server*
specifies the name or IP address of the SAS Metadata Server. You must specify either -METASERVER, -METAPORT, -METAUSER, and -METAPASS, or -METAPROFILE and -METACONNECT. This option is stored in the configuration file that is automatically created by the SAS Deployment Wizard.

-METAPORT *port*
specifies the port to use to connect to the SAS Metadata Server specified by the -METASERVER argument. This option is stored in the configuration file that is automatically created by the SAS Deployment Wizard.

-METAUSER ***user-ID***
specifies the user ID to use to connect to the SAS Metadata Server specified by the -METASERVER argument. This option is stored in the configuration file that is automatically created by the SAS Deployment Wizard.

-METAPASS ***password*** **| _PROMPT_**
specifies the password of the user specified in the -METAUSER argument. If the value of the argument is set to _PROMPT_, the user is prompted for a password. This option is stored in the configuration file that is automatically created by the SAS Deployment Wizard.

-METAPROFILE ***profile_pathname***
specifies the path name of the connection profile for the SAS Metadata Server. You must specify either -METASERVER, -METAPORT, -METAUSER, and -METAPASS, or -METAPROFILE and -METACONNECT. This option is stored in the configuration file that is automatically created by the SAS Deployment Wizard.

-METACONNECT ***connection-name***
specifies the name of the connection to use when connecting to the SAS Metadata Server. The connection must be defined in the metadata profile specified in the -METAPROFILE argument. This option is stored in the configuration file that is automatically created by the SAS Deployment Wizard.

Optional Arguments

-GRIDCONFIG ***grid-option-file***
specifies the path and filename of a file containing other SASGSUB options. The default value is sasgsub.cfg.

-GRIDFILESIN ***grid-file-list***
specifies a comma-separated list of files that need to be moved to the grid work directory before the job starts running.

-GRIDJOBNAME ***grid-program-name***
specifies the name of the grid job as it appears on the grid. If this argument is not specified, the SAS program name is used.

-GRIDJOBOPTS ***grid-provider-options***
specifies any options that are passed to the grid provider when the job is submitted. See "Supported Job Options " on page 89.

-GRIDUSER ***grid-logon-username***
specifies the user name to be used to log on to the grid, if required by the grid provider. This option is not required if the grid uses Platform Suite for SAS.

-GRIDPASSWORD ***grid-logon-password***
specifies the password to log on to the grid, if required by the grid provider. This option is not required if the grid uses Platform Suite for SAS.

-GRIDPLUGINPATH ***grid-jar-file-path1…grid-jar-file-pathN***
specifies a list of paths to search for additional grid provider JAR files. Paths are separated by semicolons and cannot contain spaces. This option is not required if the grid uses Platform Suite for SAS.

–GRIDRESTARTOK
specifies that the job can be restarted at a checkpoint.

-GRIDSASOPTS ***grid-sas-options***
specifies any SAS options that are applied to the SAS session started on the grid.

-GRIDWORKLOAD ***grid-resource-name***
specifies a resource name to use when submitting the job to the grid.

-GRIDWORKREM *shared-file-system-path*
specifies the path name of the GRIDWORK directory in the shared file system relative to a grid node. Use this argument when the machine used to submit the job is on a different platform than the grid. The path cannot contain spaces.

-LOGCONFIGLOC *logging-option-file*
specifies the path and name of a file containing any options for the SAS logging facility. SASGSUB uses the App.Grid logger name with these keys:

App.Grid.JobID
specifies the job ID as returned by the grid middleware provider.

App.Grid.JobName
specifies the job name.

App.Grid.JobStatus
specifies the job status. Possible values are Submitted, Running, or Finished.

App.Grid.JobDir
specifies the job directory name.

App.Grid.JobDirPath
specifies the full path of job directory.

App.Grid.JobSubmitTime
specifies the time that the job was submitted.

App.Grid.JobStartTime
specifies the time that the job started running.

App.Grid.JobEndTime
specifies the time that the job completed.

App.Grid.JobHost
specifies the host that ran the job.

-GRIDLIBPATH *path*
the path to the shared libraries used by the utility. This value is set in the configuration file and should not be altered. The path cannot contain spaces.

-VERBOSE
specifies that extra debugging information is printed. If this argument is not specified, only warning and error messages are printed.

SASGSUB Syntax: Ending a Job

The following is the complete syntax for ending a job on a SAS grid. The SASGSUB options can be specified in a configuration file so that they do not have to be entered manually. By default, the configuration file is named sasgsub.cfg and is stored in the current directory. The SAS Deployment Wizard automatically creates a configuration file that includes most of the required options. Enter the command on a Windows or UNIX command line.

Syntax

SASGSUB
-GRIDKILLJOB *job-id* | ALL-GRIDAPPSERVER *sas-application-server*
-GRIDLICENSEFILE *grid-enabled-license-file*
-GRIDSUBMITPGM *sas-program-file*
-GRIDWORK *work-directory* -JREOPTIONS *java-runtime-options*

-METASERVER *server* -METAPORT *port*
-METAPASS *password*-METAPROFILE *profile-name*
-METACONNECT *connection-name*
<-GRIDCONFIG *grid-option-file*>
<-GRIDUSER *grid-logon-username*> <-GRIDPASSWORD *grid-logon-password*>
<-GRIDPLUGINPATH *grid-jar-file-path*>
<-LOGCONFIGLOC *logging-option-file*> <-GRIDLIBPATH *path*><-VERBOSE>

Required Arguments

-GRIDKILLJOB *job-id* | ALL
terminates the job specified by *job-id*. If you specify ALL, all jobs are terminated.

-GRIDAPPSERVER *sas-application-server*
specifies the name of the SAS Application Server that contains the grid's logical grid server definition. This option is stored in the configuration file that is automatically created by the SAS Deployment Wizard.

-GRIDLICENSEFILE *grid-enabled-license-file*
specifies the path and filename of a SAS license file that contains the SAS Grid Manager license. The default value is "license.sasgsub" and the default location is the GRIDWORK directory. You must copy the license file for the grid control server to the GRIDWORK directory and rename the file license.sasgsub in order to match the default values for this option. This option is stored in the configuration file that is automatically created by the SAS Deployment Wizard.

-GRIDWORK *work-directory*
specifies the path for the shared directory that the job uses to store the program, output, and job information. The path cannot contain spaces. This option is stored in the configuration file that is automatically created by the SAS Deployment Wizard.

-JREOPTIONS *java-runtime-options*
specifies any Java run-time options that are passed to the Java Virtual Machine. This argument is required if the grid provider plug-in uses Java.

-METASERVER*server*
specifies the name or IP address of the SAS Metadata Server. You must specify either -METASERVER, -METAPORT, -METAUSER, and -METAPASS, or -METAPROFILE and -METACONNECT. This option is stored in the configuration file that is automatically created by the SAS Deployment Wizard.

-METAPORT *port*
specifies the port to use to connect to the SAS Metadata Server specified by the -METASERVER argument. This option is stored in the configuration file that is automatically created by the SAS Deployment Wizard.

-METAUSER *user-ID*
specifies the user ID to use to connect to the SAS Metadata Server specified by the -METASERVER argument. This option is stored in the configuration file that is automatically created by the SAS Deployment Wizard.

-METAPASS *password* | PROMPT
specifies the password of the user specified in the -METAUSER argument. If the value of the argument is set to PROMPT, the user is prompted for a password. This option is stored in the configuration file that is automatically created by the SAS Deployment Wizard.

-METAPROFILE *profile_pathname*
specifies the path name of the connection profile for the SAS Metadata Server. You must specify either -METASERVER, -METAPORT, -METAUSER, and

-METAPASS, or -METAPROFILE and -METACONNECT. This option is stored in the configuration file that is automatically created by the SAS Deployment Wizard.

-METACONNECT *connection-name*
specifies the name of the connection to use when connecting to the SAS Metadata Server. The connection must be defined in the metadata profile specified in the -METAPROFILE argument. This option is stored in the configuration file that is automatically created by the SAS Deployment Wizard.

Optional Arguments

-GRIDCONFIG *grid-option-file*
specifies the path and filename of a file containing other SASGSUB options. The default value is sasgsub.cfg.

-GRIDUSER *grid-logon-username*
specifies the user name to be used to log on to the grid.

-GRIDPASSWORD *grid-logon-password*
specifies the password to log on to the grid.

-GRIDPLUGINPATH *grid-jar-file-path1…grid-jar-file-pathN*
specifies a list of paths to search for additional grid provider JAR files. Paths are separated by semicolons and cannot contain spaces. This option is not required if the grid uses Platform Suite for SAS.

-LOGCONFIGLOC *logging-options*
specifies any options for the SAS logging facility. See "SASGSUB Syntax: Submitting a Job" on page 79 for a list of keys for the App.Grid logger.

-GRIDLIBPATH *path*
the path to the shared libraries used by the utility. This value is set in the configuration file and should not be altered. The path cannot contain spaces.

-VERBOSE
specifies that extra debugging information is printed. If this argument is not specified, only warning and error messages are printed.

SASGSUB Syntax: Viewing Job Status

The following is the syntax for using SASGSUB to view the status of a job on a SAS grid. Enter the command on a Windows or UNIX command line.

Syntax

SASGSUB
-GRIDGETSTATUS *job-id* | ALL -GRIDWORK *work-directory*
<-GRIDCONFIG *grid-option-file*> <-GRIDLIBPATH *path*><-VERBOSE>

Required Arguments

-GRIDGETSTATUS *job-id* | ALL
displays the status of the job specified by *job-id*. If you specify ALL, the status of all jobs for the current user is displayed.

-GRIDWORK *work-directory*
specifies the path for the shared directory that the job uses to store the program, output, and job information. The path cannot contain spaces. This option is stored in the configuration file that is automatically created by the SAS Deployment Wizard.

Optional Arguments

-GRIDCONFIG *grid-option-file*
specifies the path and filename of a file containing other SASGSUB options. The default value is sasgsub.cfg.

-GRIDLIBPATH *path*
the path to the shared libraries used by the utility. This value is set in the configuration file and should not be altered. The path cannot contain spaces.

-VERBOSE
specifies that extra debugging information is printed. If this argument is not specified, only warning and error messages are printed.

SASGSUB Syntax: Retrieving Job Output

The following is the syntax for using SASGSUB to retrieve the output of a job that has completed processing on a SAS grid. Enter the command on a Windows or UNIX command line.

Syntax

SASGSUB
-GRIDGETRESULTS *job-id* | ALL -GRIDWORK *work-directory*
<-GRIDRESULTSDIR *directory*>
<-GRIDCONFIG> <-GRIDLIBPATH *path*><-VERBOSE>

Required Arguments

-GRIDGETRESULTS *job-id* | ALL
Copies the job information from the work directory to the directory specified by -GRIDRESULTSDIR for the specified *job-id* or for all jobs.

-GRIDWORK *work-directory*
specifies the path for the shared directory that the job uses to store the program, output, and job information. The path cannot contain spaces.

Optional Arguments

-GRIDRESULTSDIR *directory*
specifies the directory to which the job results are moved. The default value is the current directory.

-GRIDCONFIG *grid-option-file*
specifies the path and filename of a file containing other SASGSUB options. The default value is sasgsub.cfg.

-GRIDLIBPATH *path*
the path to the shared libraries used by the utility. This value is set in the configuration file and should not be altered. The path cannot contain spaces.

-VERBOSE

specifies that extra debugging information is printed. If this argument is not specified, only warning and error messages are printed.

Part 3

Appendix

Appendix 1

Supported Job Options

The following table lists the job options that are supported by Platform Suite for SAS. You can specify these options in these locations:

- the JOBOPTS= option of the GRDSVC_ENABLE function
- the **Additional Options** field in the metadata definition for the SAS Logical Grid Server

Options specified in metadata override those specified on a GRDSVC_ENABLE statement.

Table A1.1 *Platform Suite for SAS Job Option Name/Value Pairs*

Job Option Name/Value Pairs	Explanation
exclusive=0|1	specifies whether the job runs as the only job on the grid node. 0 means that the job does not run exclusively; 1 means that the job runs exclusively. The default is 0.
host=*host*	specifies the name of the host to run the job on.
jobgroup=*job-group*	specifies the name of the job group to associate with the job.
priority=*job-priority*	specifies the user-assigned job priority. This is a value between 1 and MAX_USER_PRIORITY, as defined in the lsb.params file.
project=*project*v	specifies the name of the project to associate with the job.
queue=*queue*	specifies the name of the queue to put the job in. The default queue name is **`normal`**.
reqres="*requested-resources*"	specifies additional resource requirements.
runlimit=*time-in-seconds*	specifies the maximum amount of time that a job is allowed to run. This value is used as an absolute limit or as part of an SLA job.
sla=*service-level-agreement*	specifies the name of the service-level agreement to associate with the job.
usergroup=*user-group*	specifies the name of the user group.

For complete information about job options, see *Platform LSF Reference*.

Glossary

application server
a server that is used for storing applications. Users can access and use these server applications instead of loading the applications on their client machines. The application that the client runs is stored on the client. Requests are sent to the server for processing, and the results are returned to the client. In this way, little information is processed by the client, and nearly everything is done by the server.

authentication
the process of verifying the identity of a person or process within the guidelines of a specific authorization policy.

authentication domain
a SAS internal category that pairs logins with the servers for which they are valid. For example, an Oracle server and the SAS copies of Oracle credentials might all be classified as belonging to an OracleAuth authentication domain.

grid
a collection of networked computers that are coordinated to provide load balancing of multiple SAS jobs, scheduling of SAS workflows, and accelerated processing of parallel jobs.

grid computing
a type of computing in which large computing tasks are distributed among multiple computers on a network.

grid control server
the machine on a grid that distributes SAS programs or jobs to the grid nodes. The grid control server can also execute programs or jobs that are sent to the grid.

grid monitoring server
a metadata object that stores the information necessary for the Grid Manager plug-in in SAS Management Console to connect with the Platform Suite for SAS or other grid middleware to allow monitoring and management of the grid.

grid node
a machine that is capable of receiving and executing work that is distributed to a grid.

identity
See metadata identity.

job
a metadata object that specifies processes that create output.

load balancing
for IOM bridge connections, a program that runs in the object spawner and that uses an algorithm to distribute work across object server processes on the same or separate machines in a cluster.

logical grid server
a metadata object that stores the command that is used by a grid-enabled SAS program to start a SAS session on a grid.

logical server
in the SAS Metadata Server, the second-level object in the metadata for SAS servers. A logical server specifies one or more of a particular type of server component, such as one or more SAS Workspace Servers.

login
a SAS copy of information about an external account. Each login includes a user ID and belongs to one SAS user or group. Most logins do not include a password.

metadata identity
a metadata object that represents an individual user or a group of users in a SAS metadata environment. Each individual and group that accesses secured resources on a SAS Metadata Server should have a unique metadata identity within that server.

metadata repository
a collection of related metadata objects, such as the metadata for a set of tables and columns that are maintained by an application. A SAS Metadata Repository is an example.

metadata server
a server that provides metadata management services to one or more client applications. A SAS Metadata Server is an example.

plug-in
a file that modifies, enhances, or extends the capabilities of an application program. The application program must be designed to accept plug-ins, and the plug-ins must meet design criteria specified by the developers of the application program. In SAS Management Console, a plug-in is a JAR file that is installed in the SAS Management Console directory to provide a specific administrative function. The plug-ins enable users to customize SAS Management Console to include only the functions that are needed.

SAS Management Console
a Java application that provides a single user interface for performing SAS administrative tasks.

SAS Metadata Repository
one or more files that store metadata about application elements. Users connect to a SAS Metadata Server and use the SAS Open Metadata Interface to read metadata from or write metadata to one or more SAS Metadata Repositories. The metadata types in a SAS Metadata Repository are defined by the SAS Metadata Model.

SAS Metadata Server
a multi-user server that enables users to read metadata from or write metadata to one or more SAS Metadata Repositories. The SAS Metadata Server uses the Integrated Object Model (IOM), which is provided with SAS Integration Technologies, to communicate with clients and with other servers.

SAS Workspace Server
a SAS IOM server that is launched in order to fulfill client requests for IOM workspaces. See also IOM server and workspace.

Index

N

O

P

Q

R

S

Your Turn

We welcome your feedback.

- If you have comments about this book, please send them to **yourturn@sas.com**. Include the full title and page numbers (if applicable).
- If you have comments about the software, please send them to **suggest@sas.com**.

SAS® Publishing Delivers!

Whether you are new to the work force or an experienced professional, you need to distinguish yourself in this rapidly changing and competitive job market. SAS® Publishing provides you with a wide range of resources to help you set yourself apart. Visit us online at support.sas.com/bookstore.

SAS® Press

Need to learn the basics? Struggling with a programming problem? You'll find the expert answers that you need in example-rich books from SAS Press. Written by experienced SAS professionals from around the world, SAS Press books deliver real-world insights on a broad range of topics for all skill levels.

support.sas.com/saspress

SAS® Documentation

To successfully implement applications using SAS software, companies in every industry and on every continent all turn to the one source for accurate, timely, and reliable information: SAS documentation. We currently produce the following types of reference documentation to improve your work experience:

- Online help that is built into the software.
- Tutorials that are integrated into the product.
- Reference documentation delivered in HTML and PDF – **free** on the Web.
- Hard-copy books.

support.sas.com/publishing

SAS® Publishing News

Subscribe to SAS Publishing News to receive up-to-date information about all new SAS titles, author podcasts, and new Web site features via e-mail. Complete instructions on how to subscribe, as well as access to past issues, are available at our Web site.

support.sas.com/spn

sas | THE POWER TO KNOW®