SAS Grid Use Case

Doug Haigh
SAS/CONNECT Development
SAS Campus Drive
Cary, NC 27513

Doug.Haigh@sas.com
919-531-1716

2/1/2007

Introduction

For many years, SAS has had the ability to distribute SAS code to remote machines via the SAS/CONNECT product. SAS Grid Manager extends SAS/CONNECT to enable SAS customers to distribute their code to machines attached to a grid without needing to know anything about the remote machine and allowing the application to exploit the load balancing, monitoring and other capabilities of grid middleware. SAS Grid Manager divides its use of the grid into two areas: remote SAS code execution and grid monitoring.

Remote Execution

Remote SAS code execution is done via the SAS/CONNECT product which links a local SAS session with a remote SAS session through a proprietary interface over a number of network protocols such as TCP/IP. Before Grid Manager, the SAS code had to specify name or address of the remote machine it was planning on using. The remote machine would be running a telnet deamon or a SAS/CONNECT spawner that would start a SAS session on the remote machine. Using Grid Manager, the remote machine is determined by the grid middleware and a script file is run on that remote machine that starts SAS with the name and port of the local machine it needs to connect to.

To create a SAS session on a machine in the grid, the local SAS session needs to know where and how to submit jobs to the grid. SAS uses the concept of a logical grid server stored in metadata for this information. A logical grid server definition in metadata contains the grid middleware connection information such as grid provider, grid control server, grid username/password, and grid command to start SAS on a remote machine in the grid. This information is retrieved by SAS language processing and used along with some resource/property information to create the grid job.

SAS uses grid resources/properties to specify which machines in the grid can run the job. Machines in the grid that have SAS installed are expected to have resources/properties associated with them that indicate to the grid control server that this machine can handle SAS programs. The resource/properties are also used to specify subsets of grid nodes (called workloads) that may have special SAS modules installed that are used for specific SAS programs. The resources/properties are used as simple Boolean flags allowing the selection of grid nodes for processing. For example, under Platform LSF, these are Boolean resources that are associated with a host; under Data Synapse, these are engine properties with a value of 1; under United Devices, these are device resources with a non-zero value. Grid middleware that does not support the concept of generic Boolean resources (such as Microsoft CCS) would be expected to handle running SAS programs on every node in the grid since SAS would not be able to differentiate one grid node's capabilities from another.

With this information we require the following functions for remote SAS code execution:

- Ability to query the grid to determine number of execution nodes based on a set of resource/property names since several SAS products are grid enabled and try to break up their data or loops based on an idea of the number of nodes they could possibly run on.

  If the grid middleware does not provide this capability through its API, SAS would need to query all the nodes in the grid and determine if they are active participants in the grid (license problems or not accepting jobs would make a host inactive). Once all active hosts are determined, their resources/properties are searched for the required resource/property values.

- Ability to submit a job to the grid with the restriction of running only on nodes specified by a set of resource/property names. The job will run a shell script on a remote machine with a set of arguments. The remote job will often run as a specified user for access to required operating system resources (files, machines, etc.). The name of the job is also passed for monitoring purposes.

  Some grid middleware will allow the specification of resources on the job submission; others require you to build a candidate host list. If it is the latter, logic similar to the logic used to query the number of execution hosts is used.

- Ability to query the status of the job and return that the job is either pending, running, failed, or succeeded.

- Ability to determine the name of the host that is executing the remote session if a job is running.

  Several SAS products like to display the name of the execution host as the job is executing. This also helps in debugging problems in the grid.

- Ability to cancel execution of a currently running job or cancel a pending execution.

  This should prevent a pending job from being scheduled or terminate the process on the grid node of a currently running job. This should not remove the job from the information database, but should indicate the job was terminated by the user.

Grid Monitoring

SAS Grid Manager includes a plug-in to the SAS Management Console that is used to monitor the jobs, hosts, and queues in the grid. The following functions are required for the plug-in and must be available through Java classes:

- Ability to get a list of jobs, hosts or queues (for systems that do not have queues, this will be a list of one element)

- Ability to cancel a selected job

- Ability to get a list of property values for a job which must include at least
    - Job ID
    - Job Name
    - Status (pending, running, done, failed)
    - Submit time
    - Start time
    - End time
    - Execution host

- Ability to get a list of property values for a host which must include at least
    - Host name
    - Host IP address
    - Status (active, inactive)
        - An active host will be able to accept a job sometime in the future for processing. A host can be active, but currently unable to accept a job which is the case when a host is currently processing all the jobs it can handle.
        - Inactive hosts are hosts that are part of the grid, but for some reason cannot accept any new jobs. This can be due to communication problems, license problems, or the host was closed by the grid admin. Usually if a host is inactive, the grid admin is required to do something to make it active again.
    - Maximum number of jobs allowed to execute
    - Number of running jobs
    - Number of CPUs
    - Resource/Properties assigned

- Ability to get a list of property values for a queue which must include at least
    - Queue name
    - Total number of jobs
    - Number of running jobs
    - Number of pending jobs