

Final Report

Metascheduling Requirements Analysis Team

1 Introduction

This Requirements Analysis Team (RAT) builds on the work from the previous Scheduling RAT by refining TeraGrid metascheduling requirements, documenting deployed Resource Provider (RP) technologies and policies, and investigating metascheduling technologies. It concludes by recommending a path that TeraGrid should take in the area of metascheduling including capabilities to deploy and technologies that might be used to provide these capabilities.

Membership of this RAT included representatives from the Grid Infrastructure Group (GIG) as well as all of the RPs. We believe the inclusion of major TeraGrid stakeholders results in the outputs of this RAT being widely agreed with across TeraGrid. This will allow us to immediately make progress deploying metascheduling capabilities once the RAT completes.

Based on our experiences and results from the previous Scheduling RAT, we identified a set of capabilities that we believe might benefit TeraGrid users and structured our RAT around them. The capabilities are:

- ***Advance reservations.*** An *advance reservation* dedicates a set of resources (compute, visualization, etc.) to a user or group of users at a specific time for a specific duration. For instance, a user may need resources for a demonstration, to meet a deadline, or for simultaneous access to TeraGrid and non-TeraGrid resources.
- ***Co-scheduling.*** There are some simulations or analyses that require simultaneous use of multiple resources. For example, a user needs to run a simulation with extremely large memory requirements that cannot be met by using a single resource. By using multiple resources at the same time, *co-scheduling* enables this kind of capability.
- ***On-demand scheduling.*** Users need to execute high priority jobs quickly (such as in emergency situations). We observe two different *on-demand* capabilities:
 - ***Highest priority.*** This kind of on-demand job runs “next”. It receives the highest priority and executes as soon as resources become available, letting running jobs finish.
 - ***Preemption.*** This kind of on-demand job requires resources to be freed so that the job can run immediately.
- ***Automatic Resource Selection.*** This capability automatically selects resources for a job based on user-specified requirements and preferences. Users invoke this capability when they have a job that can run on any of a number of resources and they want to meet a scheduling goal such as minimizing turn around time.

- **Ensemble.** An *ensemble* is a set of independent jobs that may or may not run at the same time. Users wish to specify *ensembles*, manage an *ensemble* as a whole, and have scheduling goals for an *ensemble* rather than an individual job. One common type of *ensemble* is a parameter sweep where the same application is run over and over with slightly different input parameters.
- **Workflow.** *Workflows* consist of jobs with dependencies between them. For instance, a user may need a visualization to start on a resource after a computation completes on another resource.

This RAT performed four major tasks. The first task updated and documented our understanding of user metascheduling requirements via a user survey. The previous Scheduling RAT completed over a year ago and we wanted our recommendations to be based on current user needs. The second task compiled information about RP scheduling environments, requirements, and preferences. The third task evaluated metascheduling tools and identified those that should be investigated further. The final task developed recommendations based on the information we gathered from the first three tasks.

This report summarizes the information we gathered and provides our recommendations. Appendix A lists the participants in this RAT.

2 Recommendations

Based on the information we gathered and our discussions, we have developed a number of recommendations for the TeraGrid in the area of metascheduling. It is important to note that we are primarily focused on automated metascheduling where there are no humans in the loop. This type of scheduling is necessary when there are many scheduling decisions that need to be made. Humans cannot make many decisions quickly and it would therefore not be possible to have efficient metascheduling that depends on such decision making.

However, there are situations where humans must be in the loop, such as when extraordinary requests for resources are made. When this occurs, the automated tools must incorporate knowledge of these decisions. An example would be when a user asks for a reservation of a large number of resources for an extended period of time. Only a person might be allowed to grant such a request since it is out of the ordinary, but the metascheduling tools used by the user receiving the reservation must know about the reservation.

We organize our metascheduling recommendations into a set of general recommendations and then capability-specific recommendations.

General Recommendations:

We recommend that a new Metascheduling Working Group be formed to evaluate and deploy metascheduling capabilities in production on the TeraGrid. While this RAT identified important capabilities and identified potential tools and policies to provide these capabilities, a thorough hands-on evaluation of tools and policies was not in the scope of our work. We believe that a focused Working Group (WG) with representatives from all the RPs, User Services and the GIG-Pack is necessary to provide metascheduling capabilities on TeraGrid.

We recommend that TeraGrid provide common client interfaces to metascheduling capabilities. Our user survey found interest in web interfaces, command line programs, APIs in various languages, and web service interfaces. The appropriate interfaces to provide should be chosen independently for each capability, but we recommend that commonly used capabilities be available via the TeraGrid User Portal. Programs and APIs that are chosen as interfaces should be provided as part of a CTSS kit.

We recommend that for each metascheduling service that is deployed at RP sites, that the TeraGrid GIG provide the service as part of a CTSS kit and that each service

- allows the RP to set their local policies,
- provides information about these policies to TeraGrid users and their tools, and
- handles metascheduling requests with no human in the loop.

We recommend that the RPs and GIG create testbeds for evaluating metascheduling technologies with the goals of deploying technologies, interfacing the technologies to local workload managers, and providing capabilities for user feedback. The systems and personnel participating in each testbed should be selected based on user demand. Systems identified to participate in the testbeds include ANL Itanium and Viz, NCSA Mercury, PSC MrRodgers, SDCS Itanium, Datastar, and ondemand, TACC Lonestar and Maverick.

Advance Reservation Recommendations:

We recommend that TeraGrid provide advance reservation capabilities to its users as a high priority. Our survey results indicate that advance reservation is a highly requested capability from the users we surveyed. For TeraGrid to provide this capability, it must be supported by local scheduling systems.

We have identified GUR and HARC as software that may satisfy these needs and recommend that it be further evaluated.

Co-Scheduling Recommendations:

We recommend that TeraGrid support co-scheduling since our user survey indicated that it is a highly requested capability and a manual version of this capability is already being used via cross site runs. We also note that one method of providing co-scheduling is via advance reservations of multiple systems and that advance reservations were a highly requested feature in our survey. We have identified the following software that may satisfy these needs and recommend that they be further evaluated: GUR, HARC

On-Demand Recommendations:

We recommend that on-demand computing be supported on a subset of the TeraGrid. There are a few TeraGrid users that identify on-demand access to systems as a critical capability and we believe these users should be supported. TeraGrid should target the systems that these users want to use and attempt to support on-demand access to those systems.

We have identified SPRUCE as candidate software to support on-demand and recommend that it be further evaluated.

Automatic Resource Selection Recommendations:

We recommend that a metascheduler be deployed that provides automatic resource selection across as many TeraGrid resources as possible. This is a highly requested capability in our user survey and is deployable with low impact on RPs. Our technology evaluation identified a number of mature software packages that provide this capability: GridWay, GRMS, Moab.

We recommend that this capability be provided in an iterative manner. Automatic resource selection can be provided with a variety of degrees of sophistication. We believe that the best approach is to provide a simple solution quickly and then increase the sophistication of this solution rather than take more time to initially deploy a sophisticated solution.

Ensemble Recommendations:

We recommend that TeraGrid support ensemble execution. Our user surveys identified ensembles as a highly requested feature, but that a number of tools are already being used. Therefore, TeraGrid may not need to provide additional ensemble tools, but will need to provide the capabilities these tools need.

Workflow Recommendations:

We recommend that TeraGrid support workflow execution. A number of users in our user survey indicated interest in workflows, but only a few rated it critical or important and they are already using their own workflow tools. Therefore, TeraGrid may not need to provide additional workflow tools, but will need to provide the capabilities these tools need.

We recommend that TeraGrid workflow tools not require local workload managers to handle dependencies between jobs. Cluster workload managers support job dependencies at different levels so we believe the base assumption should be that local workload managers do not handle dependencies.

3 User Information

We interviewed a total of 13 TeraGrid user groups and 5 TeraGrid Science Gateway collaborations, for a total of 18 respondents. Our goal was to determine their interest in, and urgency of their need for, the six capabilities listed in Section 1. The RAT defined the interview questions in plenary session and an interview web form was provided to the TeraGrid's User Services and Science Gateways working groups. The members of that working group (TG staff) were instructed to approach the user groups they collaborate with in the ASTA and Science Gateways programs. This was done in order to target active users who have considerable hands-on experience with the existing capabilities of the TeraGrid. The user groups that responded range from a two-user research team (Chung) to representatives of very large international E-Science collaborations (Coveney and Nakano).

The interview questions and answers are summarized in Table 1. The exact questions in the survey and the answers from our responders are collected in the companion Survey Data document from the RAT. For each capability, the table shows how the number of users that were interested in a capability and, of those that are interested in a capability, how many rate the capability as critical or important and how many want the capability as soon as possible. The table also shows the number of users that need the capability on systems at four or more RP sites.

Table 1. Summary of user survey results.

	Number of Users/Gateways - Details
<i>Advance Reservation</i>	
Interested	14
Critical, Important	6, 3
Now/ASAP	7
Need \geq 4 RPs	7
Software Used	6 – TG Cross Site Web (2), Local Scheduler (4)
<i>Co-Scheduling</i>	
Interested	10
Critical, Important	4, 1
Now/ASAP	4
Need \geq 4 RPs	5
Software Used	2 – TG Cross Site Web (2)
<i>On-Demand</i>	
Interested	6
Critical, Important	1, 2
Now/ASAP	3
Need \geq 4 RPs	2
Software Used	3 – SPRUCE (1), Local Scheduler (2)
<i>Automatic Resource Selection</i>	
Interested	11
Critical, Important	2, 6
Now/ASAP	6
Need \geq 4 RPs	6
Software Used	2 – Custom (1), Wolski's queue prediction
<i>Ensemble</i>	
Interested	11
Critical, Important	3, 5
Now/ASAP	7
Need \geq 4 RPs	5
Software Used	3 – Ninf-G (1), MyCluster (1), Custom (1)
<i>Workflow</i>	
Interested	11
Critical, Important	3, 1
Now/ASAP	5
Need \geq 4 RPs	4
Software Used	4 – Application Hosting Environment (1), MyCluster (1), Condor DAGMan (1), Pegasus (1), Kepler (1), Taverna (1)
<i>User Interface</i>	
Web	5
CLI	7
GUI	3
Web Service	2
API	6 - Perl (2), Python (2), C/C++ (4), Java (4)

If any responder has used software that provided a capability, that software is listed along with the number of users that have used it. Finally, the table shows what user interfaces are desired for the capabilities including Web interface, Command Line Interface (CLI), Graphical User

Interface (GUI), Web service interface, and Application Programming Interface (API) including the programming language.

The ability to create *advance reservations* was the most requested capability with 14 of 18 survey responders interested in it. Three of the responders interested in *advance reservation* only want it as a means to provide *co-scheduling*. Four of the users want this capability so that they can perform demonstrations, meet deadlines, obtain predictable completion times, and collaborate more efficiently. Finally, seven of the responders want *advance reservation* for both *co-scheduling* and the additional reasons described above. As shown in the table, a number of users have used the TeraGrid cross-site web form but a number have also directly used the reservation capabilities available from cluster scheduling systems.

Ten responders expressed an interest in *co-scheduling* making it a frequently requested capability. Further, four of the users who desire this capability have used it or would use it to execute MPICH-G2 jobs on multiple systems. The only implementation of this capability that our survey respondents reported using is the TeraGrid cross-site reservation web form.

A few users identified problems related to the manual process behind the TeraGrid cross-site reservation procedure to request reservations. The manual nature of this process sometimes causes misunderstandings and human errors that result in cross site runs not being successful.

Six of 18 respondents expressed an interest in *on-demand* computing capacity. However, one group of TeraGrid users identified it as critical. The survey identified emergency situations and deadlines as the two reasons this capability is wanted.

Eleven users were interested in *automatic resource selection* so that they can complete their jobs more quickly. However, only three respondents have used software that provides this capability and all of them used custom software.

Ensemble support was also frequently requested with 11 of 18 responders interested in it. Seven of the users interested in this capability for executing run parameter sweeps. Three of those responders had used an *ensemble* tool with one of them using the TeraGrid-supplied MyCluster system.

Workflow capabilities were of interest to 11 responders and a relatively large fraction (4 of them) had used one or more *workflow* tools. TeraGrid currently supports two of these tools, while a number of other tools are not supported. Users are interested in these tools to manage portions tasks such as pre-processing, data movement, and post-processing.

Ten responders want to use various combinations of these capabilities. Obvious ones such as automatic resource selection with ensemble and workflow were mentioned as well as less obvious ones such as co-scheduling within a workflow.

Finally, as Table 1 shows, users desire a variety of interfaces with command line interfaces, APIs in various programming languages, and web interfaces being the most requested.

4 Resource Provider Information

To provide metascheduling capabilities requested by users, we have to ensure that these capabilities are deployable on the production systems that are part of TeraGrid. We developed a survey for the TeraGrid Resource Providers to gather information about their resource managers

and schedulers, their scheduling goals and priorities, their plan to deploy scheduling capabilities, their policies to govern those capabilities, and their concerns and foreseeable obstacles in deploying capabilities. The raw survey results are provided in the companion Survey Data document from the RAT.

Table 2. Summary of capabilities deployed on 19 TeraGrid computational resources. The number of deployments includes capabilities accessed in an automated way and accessed manually via contacting RP personnel.

Capability	Deployment Status					
	Deployed	In < 6 Months	In < 1 Year	In > 1 Year	Do Not Plan	N/A
Advance Reservation	16	1			2	
Co-Scheduling	8	1	4		5	1
On-demand (highest priority)	5	6	2		4	2
On-demand (preemptive)	1	5	4		8	1
Automatic Resource Selection	4		6	1	3	5
Ensemble	8		1	3	3	4
Workflow	4		2	5	2	6
Other						

The survey found that *advanced reservation* is deployed on 16 of the 19 systems, one resource will deploy it within one year, and there are no plans to deploy reservations on the remaining two systems. Moab is the most popular technology used to provide this capability and is used on nine systems.

The current policy on eight systems is to only allow administrators to set *advance reservations*. For a user to obtain a reservation, they must contact an administrator to request one. Nine systems expressed concern with allowing users to set their own reservations without an administrator in the loop. The RPs concerns include lack of accounting tools and rules, lack of a charging model, and the potential effect of reservations on the efficient and fair scheduling of the system.

We discussed various ways to address their concerns, namely by letting the local RP specify their local reservation policy in terms of job size, duration, frequency, and other factors. We also discussed ways to implement a different charging model for reservations including a penalty for unused reservations. These reservation policies would be set by the RP, enforced by an automated policy engine, and communicated to both the users and any automated metaschedulers. This discussion along with users strong desire for reservations influenced the RPs of an additional eight systems to agree that they could deploy an *advance reservation* capability with no administrator in the loop.

Co-scheduling is currently provided on Teragrid via cross-site runs that are requested using a web page and manually created by administrators of each local RP. It is more efficient to allow our users to create reservations themselves.

Our survey found that of 19 systems, eight reported deploying *co-scheduling*, five reported that they will have *co-scheduling* within 1 year, and five have no plan to deploy it. A variety of *co-scheduling* software is already deployed, but the most popular is GUR (on eight systems).

From the 19 systems in our survey, five report to have deployed the *highest priority on-demand* capability, eight systems plan to deploy it within 1 year, and four systems have no plan to deploy it. The RPs for the remaining two systems are uncertain about their plans for deploying this capability. The *preempting* version of *on-demand* computing is deployed on one system, is planned for nine systems, and there are no plans to deploy it on eight systems. The most used software for *on-demand* scheduling is SPRUCE.

Many sites were very concerned about *on-demand* scheduling with *preemption*. The main concern is the effect it could have on other users' jobs. Queued jobs could be significantly delayed if there are many on-demand jobs and running jobs that are preempted may need to have allocation reimbursed and users notified about the preemption for recovery. During discussion, several policies for *on-demand* scheduling emerged to minimize its impact. One policy is to only allow selected users to submit on demand jobs. Another policy is to restrict the frequency of *on-demand* jobs. A final policy is to charge *on-demand* jobs at a higher rate than lower-priority jobs.

Four of the 19 systems in our survey have an *automatic resource selection* capability. RPs plan to install it on seven systems within one year or more. This capability will not be deployed on three systems and deployment plans are undetermined on the five other systems.

The most popular software for *automatic resource selection* is Moab and is available on six systems. However, this approach requires that RPs adopt Moab as their cluster scheduler. Responders from eight resources reported that they are concerned about this capability while responders from remaining 11 resources are not concerned. The concerns about this capability are primarily due to determining local scheduling policies when there are "grid jobs" and "local jobs" as well as some concern about the unknown consequences of deploying this new capability.

Eight of nineteen systems already support *ensembles*. Seven systems will have it within a year or more, three systems don't plan to deploy it, and it is undetermined when four systems will have this capability. The most popular software is Gridshell, deployed on three resources.

Responders from four systems expressed concern about this capability while the remaining responders are not concerned. The main concern is the effect this capability could have on other jobs. A potential policy to mitigate this concern is to limit the amount of jobs submitted from a single user.

Four systems already have a *workflow* capability. Two systems will have this capability within a year, five will have it in more than a year, there are no plans to deploy it on two of the systems, and no decision has been made for the last six systems. At this time, there is no software package that is used significantly more than others and most of the *workflow* tools in use are developed by a particular science gateway.

Responders from six systems are concerned about this capability while the remaining 13 responders were not concerned at all about this capability. The main concern is related to the software development effort needed to support this capability when it isn't supported by the local workload manager. Other concerns are issues with resource availability in the job *workflow* and effects on other users' jobs.

5 Available Metascheduling Technologies

This section contains an evaluation of metascheduling technologies that exist today and that provide the metascheduling capabilities outlined in the introduction. The intention of this evaluation is to collect enough information about each technology to be able to make a preliminary selection of technologies to recommend for a later detailed evaluation and selection for use on the TeraGrid. The list of criteria used to evaluate each technology as well as the technology evaluations are provided in the companion RAT Survey Data document.

5.1 Evaluation Methodology

We gathered information about these technologies primarily via product documentation (websites, technology papers, product manuals, etc.). The ensuing detailed evaluation of a subset of these technologies should include an architectural evaluation, installation, and evaluation in a testbed with user feedback.

The following methodology used in the evaluation:

1. Through discussion on the weekly conference calls, the RAT compiled a set of questions as the evaluation criteria. The questions were designed to help evaluate the suitability of the scheduling technology for use in the TeraGrid based on factors such as technical capabilities, software dependencies, and deployment costs.
2. Members of the RAT proposed current scheduling technologies to investigate. The list contained fourteen technologies with two of them dropped during the initial phases of the evaluation at the request of the technology developers.
3. Each technology on the list was assigned to one individual RAT member for evaluation. The assignment was made so that RAT members did not evaluate a technology if they were part of the development team for that technology.
4. The completed evaluations were circulated to the full RAT membership for comments, and the evaluator addressed any resulting issues and questions. The final set of evaluations can be found in the Survey Data document produced by the RAT.
5. The RAT reviewed the evaluations and recommended a subset of scheduling technologies for detailed evaluation.

5.2 Evaluation Criteria

The evaluation criteria used can be broadly classified into three sections:

- The capabilities provided by a technology.
- Requirements of the scheduling technology, which will impact the willingness of the Resource Providers to support the technology. The dependencies include
 - a. software required on each grid resource, or grid-wide services,
 - b. the ability of the scheduler to interface with supported TeraGrid local resource managers (LRMs) such as PBS, LSF and Sun Grid Engine, and
 - c. operational dependencies impacting the current local configuration and policies of the LRM such as allowing local users to submit jobs directly to the local LRM.
- Factors affecting deployment costs for TeraGrid and its Resource Providers, including
 - a. product costs and licensing terms,
 - b. production-readiness of current release of the technology,
 - c. support and documentation available from the developer, and
 - d. availability of source code and mechanisms for developer to accept code changes from TeraGrid.

The full version of the evaluation criteria questions can also be found in the Survey Data document produced by the RAT.

Table 3: List of evaluated schedulers.

Identifier	Information
CSF	Name: Community Scheduling Framework Developer: Platform Computing & community URL: http://www.globus.org/toolkit/docs/4.0/contributions/csf/
Condor-G	Name: Developer: University of Wisconsin URL: http://www.cs.wisc.edu/condor/condorg/
GEMLCA	Name: Grid Execution Management for Legacy Code Architecture Developer: University of Westminster URL: http://www.cpc.wmin.ac.uk/gemlca/
gLite	Name: gLite Workload Management System Developer: URL: http://glite.web.cern.ch/glite/wms/
Gridbus	Name: Grid Service Broker Developer: University of Melbourne URL: http://www.gridbus.org/broker/
GRMS	Name: Grid Resource Management System Developer: Poznan Supercomputing and Networking Center (PSNC) URL: http://fury.man.poznan.pl/gridge/
Gridway	Name: Gridway Metascheduler Developer: Universidad Complutense de Madrid URL: http://www.gridway.org/
GUR	Name: Grid Universal Remote Developer: San Diego Supercomputing Center URL: http://www.sdsc.edu/~kenneth/gur.html
HARC	Name: Highly-Available Robust Co-allocator Developer: LSU Center for Computation & Technology URL: http://www.cct.lsu.edu/personal/maclaren/CoSched/
HPC Synergy	Name: HPC Synergy Developer: United Devices, Inc. URL: http://www.ud.com/products/hpcsynergy.php
MARS	Name: Michigan Advanced Resource Scheduler Developer: University of Michigan URL: http://www.mgrid.umich.edu/projects/mars.html
Moab	Name: Moab Grid Manager/Moab Workload Manager Developer: Cluster Resources, Inc URL: http://clusterresources.com/
MCP	Name: Developer: San Diego Supercomputing Center URL: TBD
SPRUCE	Name: Special Priority and Urgent Computing Environment Developer: Argonne National Laboratory URL: http://spruce.teragrid.org/index.php?page=home

5.3 Evaluation Summary

5.3.1 Technology List

Table 3 contains a list of technologies selected for evaluation, along with contact information for each. At the beginning of the evaluation, it was determined that two of the schedulers, MARS and GEMLCA, were not suitable for production deployment. These schedulers were not considered any further in this evaluation.

5.3.2 Technology Capabilities

The tables in this section evaluate the capabilities of each proposed technology. Table 4 captures the technologies ability to support *advanced reservations* and *co-scheduling*. *Co-scheduling* typically uses the advanced reservation feature of the underlying local resource manager. The table indicates that the only technologies that provide general support for these features are GUR, HARC, and Moab.

Table 4: Advanced reservations and co-scheduling.

Name	Advanced Reservations	Co-Scheduling
CSF	Yes (LSF only)	Yes
Condor-G	No	No
gLite	No, but planned	No
Gridbus	No	No
GRMS	No	No
Gridway	No	No
GUR	Yes	Yes
HARC	Yes	Yes
HPC Synergy	No	No
Moab	Yes	Yes
MCP	No	No
SPRUCE	No	No

Table 5 lists the ability of the technologies to support *automatic resource selection*, *ensembles*, and *workflows*. The schedulers in the list that provide automatic resource selection include Condor-G, gLite, GRMS, Gridway, HPC Synergy and Moab. Of the list of schedulers evaluated, Condor-G, gLite, GRMS, Gridway and Moab support both ensemble jobs as well as user specified workflows.

Table 5: Automatic resource selection, ensembles, and workflows.

Name	Automatic Resource Selection	Ensemble Jobs	Workflows
CSF	Yes	No	No
Condor-G	No, but easy to add	Yes	Yes
gLite	Yes	Yes ¹	Yes
Gridbus	Yes	Yes ²	No
GRMS	Yes	Yes	Yes
Gridway	Yes	Yes	Yes
GUR	No	No	No
HARC	No	No	Yes
HPC Synergy	Yes	No, but planned	Yes
Moab	Yes	Yes	Yes
MCP	Yes ³	No	No
SPRUCE	No	No	No

Table 6 lists the schedulers that support *on-demand* scheduling for high-priority user jobs. SPRUCE, MCP, Moab, GUR and Condor-G support this feature, but each of them required support for on-demand scheduling from the local resource manager.

Table 6: On-Demand scheduling.

Name	On-Demand	
	Priority	Pre-emptive
CSF	No	No
Condor-G	Yes	Yes
gLite	No	No
Gridbus	No	No
GRMS	No	No
Gridway	No	No
GUR	Yes	Yes
HARC	No	No
HPC Synergy	No	No
Moab	Yes	Yes
MCP	Yes	Yes
SPRUCE	Yes	Yes

Table 7 addresses the ability of the technologies to support flexible scheduling policies. We expect that the TeraGrid GIG will require the ability to customize metascheduling policies to attempt to meet TeraGrid wide, science gateway, individual user, and resource provider goals.

¹ Requires Workload Proxy Manager component

² Parameter sweeps only

³ Submits job to all resources and subsequently cancels all but one.

Table 7: Scheduling policies.

Name	Flexible Scheduling Policies
CSF	Yes, primitive but extensible framework
Condor-G	Yes, job & user priorities; fair share
gLite	Yes, but no priorities
Gridbus	To some degree
GRMS	Yes, can add custom modules
Gridway	Yes
GUR	N/A
HARC	N/A
HPC Synergy	Yes, priorities, load, fair-share, QoS/SLA
Moab	Yes, priorities, load, fair-share, QoS/SLA
MCP	No
SPRUCE	Yes

It is clear from the tables in this section, that no single solution will meet all TeraGrid metascheduling requirements. Rather, the solution is likely to include several tools that are configured to inter-operate and provide the required overall functionality.

Table 8: Impact on local resource manager.

Name	LRMs supported	LRM Modifications?	Direct Submission
CSF	GRAM(WS&Pre-WS), LSF	No	Yes
Condor-G	GRAM(WS&Pre-WS)	No	Yes
gLite	GRAM(WS&Pre-WS)	No ⁴	Yes
Gridbus	GRAM(WS&Pre-WS), OpenPBS, SGE	No	Yes
GRMS	GRAM(WS&Pre-WS)	No	Yes
Gridway	GRAM(WS&Pre-WS)	No	Yes
GUR	Catalina, Moab, LSF, Simon	No	Yes
HARC	PBSPRO	No	Yes
HPC Synergy	GRAM(Pre-WS)	No	Yes
Moab	GRAM(WS&Pre-WS), Torque,PBS,LSF,LoadLeveler	Yes; integration with Moab monitor daemon	Yes
MCP	All ⁵	No	Yes
SPRUCE	GRAM (Pre-WS), PBS, LSF, LoadLeveler	No ⁶	Yes

⁴ Jobs run as VO, not as individual user

⁵ Might be necessary to add a stanza to a configuration file with submit/query syntax

⁶ Needs a special queue and policies implemented by the LRM.

5.3.3 Technology Dependencies & Requirements

This section evaluates the technologies to determine how easily they can be integrated into the existing TeraGrid infrastructure. The ability to integrate the scheduler with existing local resource managers such as PBS, LSF, and SGE are given in Table 8. Factors considered are

- scheduler support for job submission/management to all LRMs currently deployed on TeraGrid resources,
- changes required to the LRM at a TeraGrid site in order to integrate the scheduler, and
- whether the scheduler restricts the usage of the resource by (possibly local) users by submitting jobs directly to the local LRM.

As shown in the table, most technologies other than GUR, HARC, Synergy and SPRUCE support both Pre-WS and WS GRAM. All the technologies require little to no modifications to the LRMs, and do not restrict direct usage of the resource by users.

Another factor in technologies' suitability for the TeraGrid is their dependency on other software packages that are not part of the current TeraGrid CTSS. Table 9 shows the additional software required by each technology on each TeraGrid system, on the machine running the metascheduling service, and any other software services required elsewhere on the grid.

Most of the technologies have dependencies on standard grid software such as Globus and Condor. The *automatic resource scheduling* capability typically depends on other services to provide monitoring information about the local resources. Most of the technologies use Globus/MDS information services and typically only require a customized agent on the local resource. However, there are some technologies (such as gLite) that have dependencies on other specific information services packages that are not currently part of the CTSS.

Table 9: Additional software requirements.

Name	On Each Node	On Metascheduler	On a grid system
CSF	GRAM	GT4 container, GRAM clients	None
Condor-G	Globus (GRAM, GSI, GridFTP), VDT Scalability patches	Globus	None
gLite	Globus (GRAM GSI), Condor, gLite WM client, R-GMA client or MDS	RH Linux 3.0 Apache httpd	Other gLite components, eg. Logging/Bookkeeping, Data Management, Information system services
Gridbus	None	Globus container	None
GRMS	Globus (GRAM, GridFTP, GIIS/GRIS), Gridge monitoring client (optional)	GRMS WS and Broker, RDBMS (Mysql or PostgreSQL)	MDS (optional) SRB (optional)
Gridway	Globus	Globus (GRAM, GridFTP, MDS)	None
GUR	GUR agent, Globus, GPFS (optional)	No central scheduler, GUR client required on submit node	None
HARC	HARC agent	No central scheduler, HARC client required on submit node	None
HPC Synergy	Synergy Client	RH Linux	HPC Insight (optional)
Moab	Moab Monitor daemon	Linux/Unix variant	None
MCP	None	None	None
SPRUCE	Globus or local scripts	No central scheduler, job script template required on submit node	Portal for administration and authorization

Another important evaluation criterion is the set of client interfaces provided by each scheduler. Since it is expected that TeraGrid users will often use higher-level tools such as application portals and not interact directly with the scheduler, it is critical that the schedulers adopted by TeraGrid have a variety of client interfaces that facilitate the development of higher-level tools.

Table 10: Client interfaces supported.

Name	Web Services	Command Line	Client APIs	Custom Tools
CSF	Yes	Yes	Java	Portlet
Condor-G	No	Yes	No	None
gLite	Yes	No	Yes	None
Gridbus	No	Yes	Java	Desktop GUI
GRMS	Yes	Yes	No	Portal
Gridway	No	Yes	Yes	None
GUR	No	Yes	No	None
HARC	No	No	No	Portal
HPC Synergy	Yes	Yes	No	Browser-based GUI
Moab	Yes	Yes	C & Java	Desktop GUI
MCP	No	Yes	No	None
SPRUCE	Yes	No	No	Browser-based GUI

As shown in Table 10, the technologies support interfaces ranging from command line, to APIs, to web service interfaces. Additionally, several technologies provide browser-based or custom GUI tools for their users.

5.3.4 Deployment Considerations

Deployment cost is an important consideration when selecting a new metascheduling technology for TeraGrid. Deployment costs are directly dependent on a number of factors including the cost of licensing and support from the scheduler developer. Since the metacheduling technologies are still evolving, it is likely that TeraGrid will require changes to the technology, which may lead to more costs. This may be of concern, especially if the code is not open-source, and if the developer does not have a process in place to accept code changes made by TeraGrid. These factors are evaluated in Table 11.

Table 11: Licensing costs and code availability.

Name	Cost/Licensing		Code Availability	
	Software	Support	Open Source	Changes Accepted
CSF	Free	Free	Yes	Yes
Condor-G	Free	Free	Yes	Yes
gLite	Free	Free	Yes	No
Gridbus	Free	Free	Yes	Yes
GRMS	Free	Free	Yes	Unknown
Gridway	Free	Free	Yes	Yes
GUR	Free	Free	Yes	Yes
HARC	Free	Free	Yes	Yes
HPC Synergy	Commercial	Commercial	No	No
Moab	Commercial ⁷	Commercial ⁷	No ⁸	No
MCP	Free	Free	Yes	Yes
SPRUCE	Free	Free	Yes	Yes

⁷ TeraGrid-wide commercial license already in place.

⁸ Might be possible to have agreement to share code.

Most of the schedulers are free and open source, except for Synergy and Moab, which are commercial products. Moab is already licensed to TeraGrid, and has indicated a willingness to work with TeraGrid to meet our requirements.

Another factor affecting the overall cost of deployment is the cost and quality of support from the developer, and the product documentation available to TeraGrid. Table 12 evaluates the schedulers in terms of the types of support available, and the quality of the product documentation.

Table 12: Documentation and support.

Name	Forms of Support				Documentation
	Email	Forum	FAQ	3 rd party	
CSF	Yes	Yes	No	No	Minimal
Condor-G	Yes	Yes	Yes	For fee	Production
gLite	??	??	??	No	Production
Gridbus	No	No	Yes	No	Production
GRMS	Yes	No?	No?	No	Production
Gridway	Yes	No	Yes	For fee	Production
GUR	Yes	No	No	No	Minimal
HARC	Yes	No	No	No	Minimal
HPC Synergy	Yes	Yes	Yes	No	Production ⁹
Moab	Yes	Yes	Yes	No	Production
MCP	Yes	No	No	No	Minimal
SPRUCE	Yes	No	Yes	No	Website

As expected, most of the open source products use email as their primary form of support. Out of the set of schedulers evaluated, Condor-G, Synergy and Moab provided the most forms of support, although support for Synergy and Moab is not free. All the schedulers evaluated had production quality support, with the exception of CSF, GUR, HARC, and MCP.

Another factor indicating the deployment readiness of the schedulers is product maturity (Table 13). The table lists the year the product was first available, the current status of the product, the number of known deployments of the schedulers in production grid environments, and the estimated number of developers supporting the product.

For general purpose, TeraGrid-wide, metascheduling services, it is probably desirable to use a technology that is in production, has been previously deployed in large grid environments, and is backed by sufficient developers to ensure that the product will evolve with emerging grid standards.

⁹ Available only to customers

Table 13: Product maturity.

Name	Available Since	Product Status	Current Deployments	Number of Developers
CSF	2004	Alpha	Unknown	~7
Condor-G	1986	Production	> 4	~25
gLite	2004	Production	1 (EGEE)	??
Gridbus	2004	Production	5 departments	~6
GRMS	2003	Production	~5	Unknown
Gridway	2005	Production	5-10	6
GUR	2004	Beta	1 (TeraGrid)	2
HARC	2005	Prototype	0	2-3
HPC Synergy	2003	Production	1-3	~20
Moab	1998	Production	Unknown	~7
MCP	2003	Prototype	0	2
SPRUCE	2006	Production	1 (TeraGrid)	6

5.4 Summary of Metascheduling Technologies

Based on the evaluations in the previous sections, it is clear that no single scheduling solution can meet all the TeraGrid user requirements for scheduling. Rather, the solution is likely to include several technologies that are configured to inter-operate and provide the required overall functionality.

The evaluations showed that GUR, and HARC should be considered for *advanced reservations* and *co-allocations*. SPRUCE, GUR and MCP are good candidates to be considered for *on-demand* support.

Although several of the technologies support *automatic resource selection*, *ensemble* scheduling and user-defined *workflows*, GridWay, GRMS, and Moab seem to provide the best overall solutions when considering their features along with other factors such as software requirements, deployment and support costs, and overall production readiness. These schedulers should be evaluated further with a hands-on study to analyze their ability to support typical TeraGrid scheduling use cases. The evaluation also shows that Moab appears to have support for most of the required TeraGrid metascheduling features, including *advanced reservations*, *co-scheduling*, and *on-demand* scheduling. If sites are willing to adopt it as their local scheduler, Moab could provide a single, integrated scheduling solution for TeraGrid.

Table 14: Summary of Metascheduling Technologies

Scheduler	Comments
CSF	Minimal, but extensible metascheduling framework; Could be extended to make it more general, but with a big development cost; not production ready, with minimal support.
Condor-G	Widely deployed, robust system. Currently no information to support metascheduling via matchmaking, and does not support grid-wide user and job priorities.
gLite	Very flexible and robust, but dependency on other gLite components
Gridbus	Well designed system, but targeted to parameter sweep jobs.
GRMS	Very flexible and supports creation of customized scheduling algorithms; integrates well with other grid components; some concerns about on-going funding for development/support. Recommended for further evaluation.
Gridway	Very flexible; good support for grid standards (e.g., DRMAA) and Globus components. Recommended for further evaluation.
GUR	Promising tool supporting <i>advanced reservations</i> and <i>co-allocation</i> . Currently in use on TeraGrid. Recommended for further evaluation.
HARC	Supports <i>advanced reservations</i> and <i>co-allocation</i> ; Can schedule to non-computation nodes (e.g. Access Grid nodes, network devices etc.). Recommended for further evaluation.
HPC Synergy	Good tools, support and documentation, but requires purchase of commercial license, and source code is not available to TeraGrid for customization. Synergy also supports desktop grids.
Moab	Has widest range of scheduling features out of all the schedulers; production quality code, support and documentation; some integration might be required, but vendor is willing to work with TeraGrid sites. Recommended for further evaluation.
MCP	A user level tool that guarantees that the user's job starts as soon as possible; not production ready, but developed by TeraGrid member site. Recommended for further evaluation.
SPRUCE	Support for <i>on-demand</i> scheduling; currently in use on TeraGrid. Recommended for further evaluation.

A. Participants

Leaders: Patricia Kovatch (SDSC, pkovatch@sdsc.edu)
Warren Smith (TACC, wsmith@tacc.utexas.edu)

GDO Supervisor: Kelly Gaither (TACC, VDIS AD, kelly@tacc.utexas.edu)

ESC member: Ralph Roskies (PSC, roskies@psc.edu)

Ashok Adiga TACC, adiga@tacc.utexas.edu
Matt Allen Indiana, malallen@indiana.edu
Bill Barth TACC, bbarth@tacc.utexas.edu
Nicholas Karonis NIU/ANL, karonis@niu.edu
Doru Marcusiu NCSA, marcusi@ncsa.uiuc.edu
Martin Margo SDSC, mmargo@sdsc.edu
Stuart Martin ANL, smartin@mcs.anl.gov
Sean McCreary NCAR, seanm@ucar.edu
Suman Nadella ANL, snadella@mcs.anl.gov
J.P. Navarro ANL, navarro@mcs.anl.gov
Greg Pike ORNL, pikeg@ornl.gov
Rich Raymond PSC, raymond@psc.edu
Sergiu Sanielevici PSC, sergiu@psc.edu
Jenny Schopf ANL, jms@mcs.anl.gov
Preston Smith Purdue, psmith@purdue.edu
Chad Vizino PSC, vizino@psc.edu
Nancy Wilkins-Diehr SDSC, wilinsn@sdsc.edu
Kenneth Yoshimoto SDSC, kenneth@sdsc.edu