DRMAA: Distributed Resource Management Application API

Andreas Haas, Sun
Hrabri Rajic, Intel

GGF 13 DRMAA session
Seoul, March 16, 2005

Agenda

- First things first
  - GGF IP
  - Sign-up sheet
  - Note takers
- Introduction
- Status of DRMAA implementations
- GridWay DRMAA implementation
- Status of DRMAA documents
- Open floor, open issues

DRMAA Charter

- Develop an API specification for the submission and control of jobs to one or more Distributed Resource Management (DRM) systems.
- The scope of this specification is all the high level functionality which is necessary for an application to consign a job to a DRM system including common operations on jobs like termination or suspension.
- The objective is to facilitate the direct interfacing of applications to today's DRM systems by application's builders, portal builders, and Independent Software Vendors (ISVs).

DRMAA history

- BOF at GGF 3 in Frascati, Oct 2001
- WG status at GGF 4, Toronto, February 2002

- Participation from PBS, SGE, Intel, LoadLeveler, Condor, Cadence, Globus GRAM
- Sideline engagement from EnFuzion, Entropia, LSF, GridIron, UD

03 Jul: Close public comment Jun

04 1H: 2 Reference implementation prototypes:

C implementations UofW Condor, Sun's SGE

CPAN Perl DRMAA-C module

Sun's SGE Java

DRMAA over Globus: GridWay project

Feedback from reference implementations fed back into spec.

04 Jun: DRMAA recommendation document accepted by GFSC

In a Nutshell

- DRMAA scope and purpose:
  - Submit, control & monitor, and query status of jobs.
  - DRMAA library could be implemented on top on OGSA and DRM systems.

- **Weekly con calls**
  - Toll Free: (866)545-5198 Code: 6898552
  - Regular: (865)521-8904

- E-mail: drmaa-wg@gridforum.org

- **Archive: http://www-unix.gridforum.org/mail_archive/drmaa-wg/threads.html**

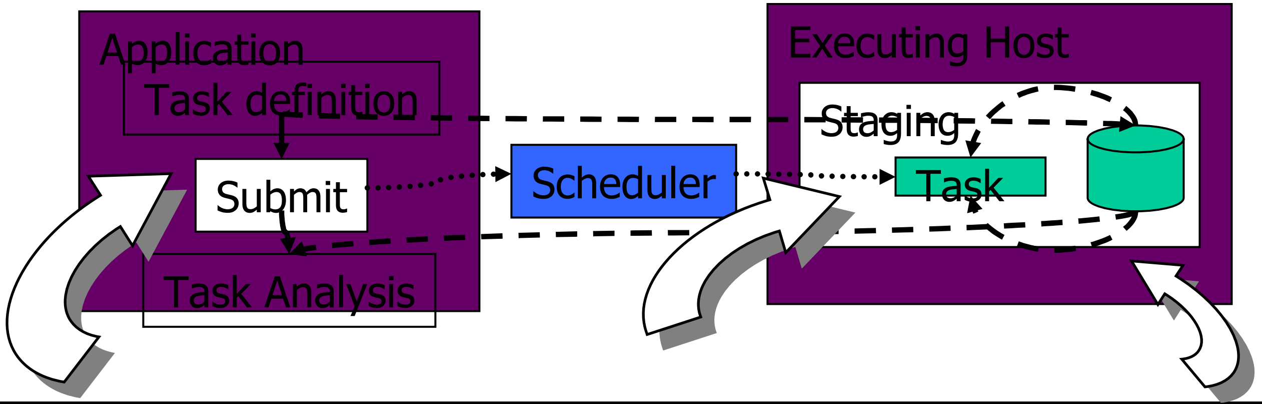
DRMAA is a Third Type of Parallelism

- PThreads and Windows threads: 1 node or SMP
- OpenMP: SMP directive based

- MPI/PVM: cluster messaging API
- ClusterMP: OpenMP on cluster

- DRMAA: cluster DRM system abstraction API
- Grid solutions
  - Globus Toolkit GRAM
  - CoG
  - UNICORE
  - GAT/SAGA
  - GridRCP solutions
  - Grid web services (OGSA)

Resource Management Systems Differ Across Each Component



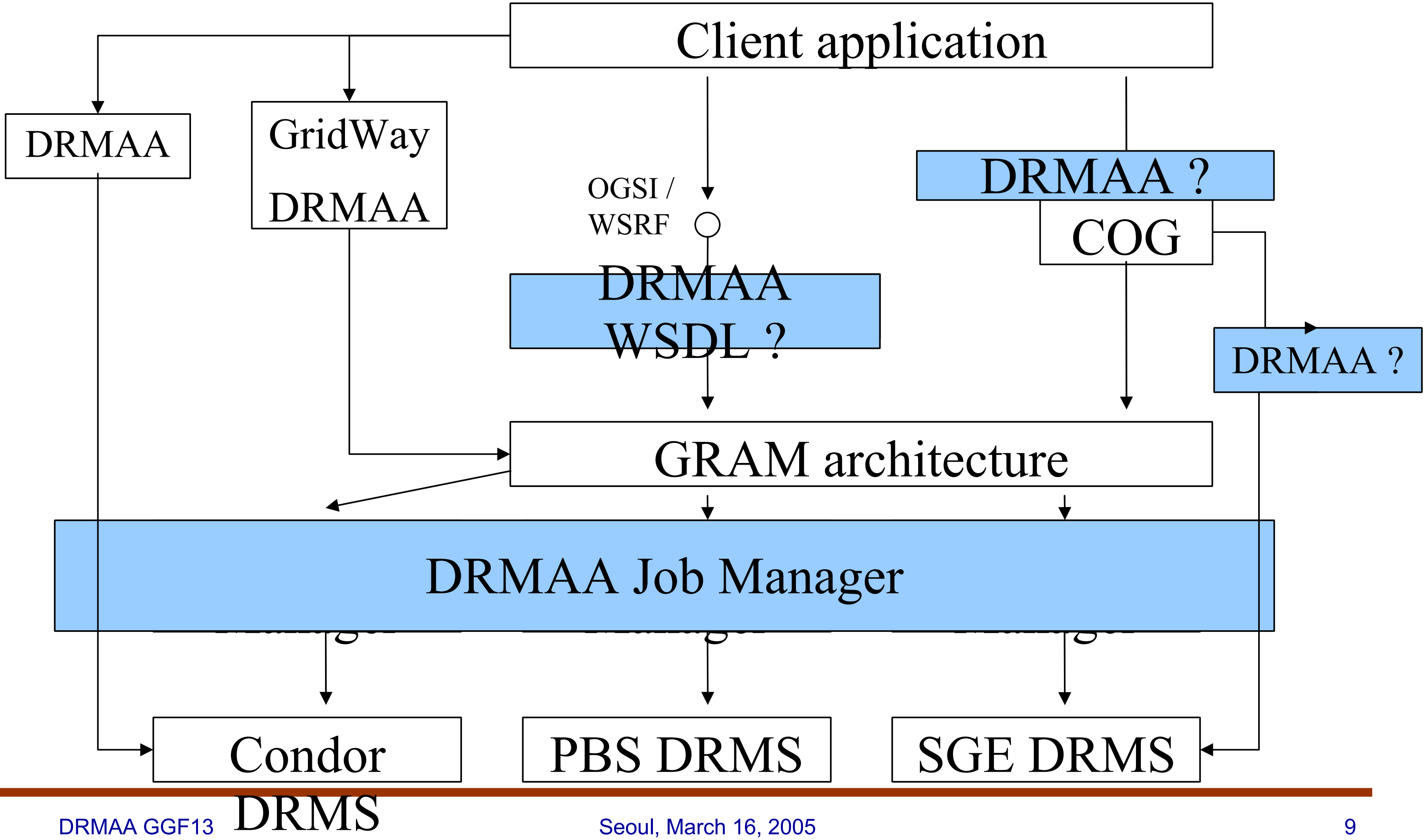
	Interface Format	Execution Environment	Platform Mix
LSF	Has API plus Batch Utilities via "LSF Scripts"	User: Local disk exported System: Remote initialized (option)	Unix ←/ → Windows
Grid Engine	GDI API Interface plus Command line interface	System: Remote initialized, with SGE local variables exported	Unix only
PBS	API (script option) Batch Utilities via "PBS Scripts"	System: Remote initialized, with PBS local variables exported	Unix ←→ Windows
DataSynapse	Proprietary API.	User: Remote initialized	Unix ← → Windows

Scope: Run a Job API

(Steps from: Ten Actions when SuperScheduling”, GGF SchedWD 8.5, J.M. Schopf, July 2001)

- **Phase 1: Resource Discovery**
  - Step 1 Authorization Filtering
  - Step 2 Application requirement definition
  - Step 3 Minimal requirement filtering
- **Phase 2 System Selection**
  - Step 4 Gathering information (query)
  - Step 5 Select the system(s) to run on
- **Phase 3 Run job**
  - Step 6 (optional) Make an advance reservation
  - **Step 7 Submit job to resources**
  - Step 8 Preparation Tasks
  - **Step 9 Monitor progress (maybe go back to 4)**
  - **Step 10 Find out Job is done**
  - Step 11 Completion tasks

DRMAA Placement



What have been the Issues?

- **Language bindings**
  - C/C++
  - Perl, Python
  - Fortran, Java
- **General features**
  - DRMAA sessions
  - Asynchronous job monitoring
  - Protocol based
  - Scalability
  - Wide characters

- **Libraries**
  - Serial / thread safe
  - Tracing / diagnosis
- **Advanced features**
  - Debugging support
  - Data streaming
  - Security
  - Categories

API groups

- Init/exit
- Job template interfaces
  - Allocate/delete
  - Setter/getter job template routines
- Job submit
  - Individual jobs
    - One time
    - Multiple times – templates (version 2)
  - Bulk jobs, implicit parameterization
- Job monitoring and control
- Auxiliary or system routines
  - trace file specification
  - error message routines
  - informational interfaces

Job Template (DRMAA JSDL)

- Functions to create/delete job template
  - job_template *drmaa_allocate_job_template (void)
  - void drmaa_delete_job_template (job_template *jt)
- Setter/getter job template routines
  - int drmaa_set_attribute(job_template *jt, char *name, char *value);
  - int drmaa_set_vector_attribute(job_template *jt, char *name, char **values);
  - char* drmaa_get_attribute(job_template *jt, char *name);
  - char** drmaa_get_vector_attribute(job_template *jt, char *name);

Job Submission

- Jobs submitted to the DRM system are identified via a job identifier
- For flexibility reasons a job identifier should be of type char *
- Single job identifiers are returned by
  - int drmaa_run_job(job_template *jt, char *job_id)
- Bulk job submissions return multiple job identifiers
  - int drmaa_run_bulk_job(char **job_ids, job_template *jt, int start, int end, int incr)

Job Monitoring, Control, and Status

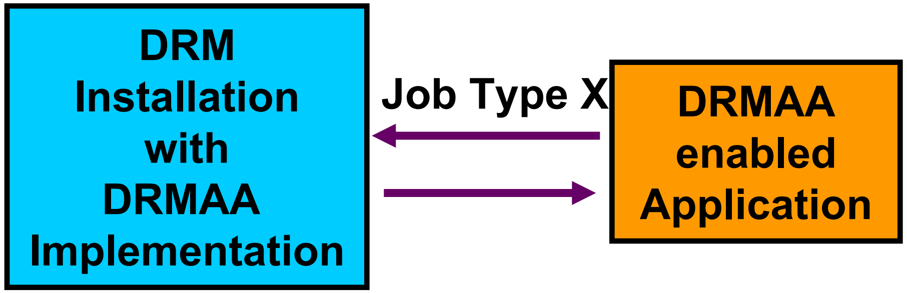
- Monitoring/Control functions
  - int drmaa_control(char *job_id, int action);
  - int drmaa_synchronize(char **job_ids);
  - int drmaa_job_ps(char *job_id, int *remote_ps);
- Blocking and non-blocking waiting for one or more jobs to finish (like wait4(2))
  - char *drmaa_wait(char *jobid, int *status, int timeout, char **rusage);
  - Use Posix functions drmaa_wifexited, etc. to get more information about failed jobs.

Native DRMS Options

- The end user interacts with the DRMS via native_resource_options parameter.
  - Simple solution
  - DRMAA implementation ignores the DRMAA DRMS implicitly used and disallowed options
  - Dist. Appls. Developers and DRMS vendors are not involved in the local environment spec.
  - The burden is on the end users to define the execution environment
    - Need to know DRM
    - Need to know the remote application installation

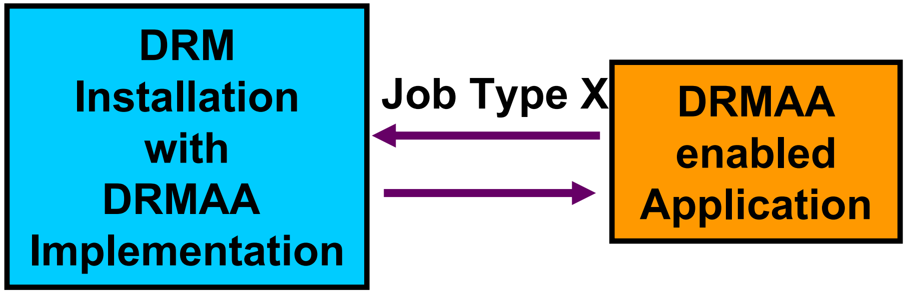
Job Categories

Site A


DRM Installation with DRMAA Implementation
Job Type X
DRMAA enabled Application

- Cluster consists of machines where X jobs run and others where they don’t run

Site B


DRM Installation with DRMAA Implementation
Job Type X
DRMAA enabled Application

- X jobs run at all machines in cluster

DRMAA Implementations

- C implementations
  - UofW Condor
  - Sun’s SGE
- CPAN Perl DRMAA-C module
  - Works only for SGE DRMAA C implementation
- Sun’s SGE Java implementation
  - 6.0u3 SGE implements C binding v0.5
- GridWay OGSA/Globus implementation
  - Subset already released, Jan 2005

DRMAA Documents Status

- DRMAA GFD-P-R or GFD.22 document
  - Since June 2004
- C binding v1.0
  - Ready for submission to GFCS
- C binding experimental v0.98
- Java binding 0.6.1
  - Fairly complete
- .NET binding v0.2
  - Needs a synch with OO v.03 IDL
- OO recommendation document
  - v0.3 and feature complete
  - Need to augment with the DRMAA GFD-P-R text
  - Will be submitted as a standalone GFD-P-R doc

DRMAA Documents Status

- DRMAA GFD-P-R or GFD.22 document
  - Since June 2004
  - To be resubmitted before GGF 14
- C binding v1.0
  - Ready for submission to GFCS
- C binding experimental v0.98
  - Has improvements that will not be rolled into DRMAA FGD-P-R doc
- Java binding 0.6.1
  - Fairly complete
- .NET binding v0.2
  - Needs a synch with OO v0.3.1 IDL
- IDL recommendation document
  - Targeting OO languages
  - v0.3.1 and feature complete
  - Need to augment with the DRMAA GFD-P-R text
  - Will be submitted as a standalone GFD-P-R doc

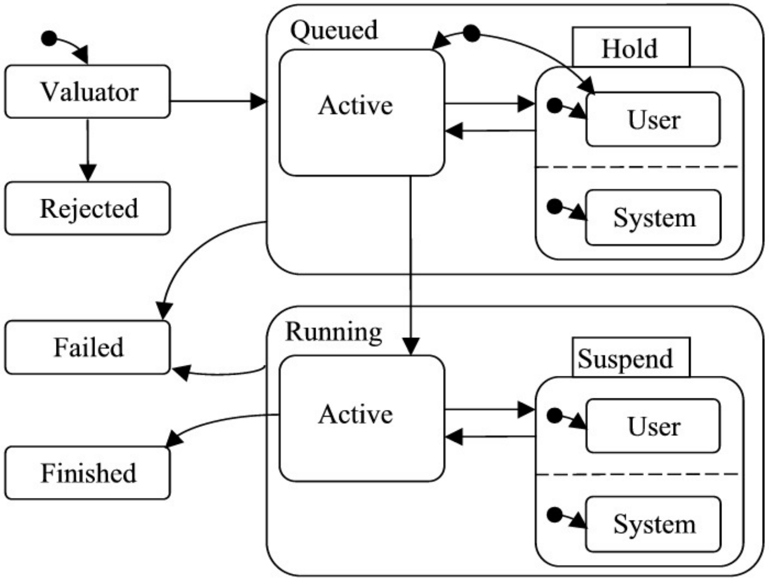
Changes: Job Info & New Attribute

- interface JobInfo {
- readonly attribute string jobId;
- readonly attribute Dictionary resourceUsage;
- readonly attribute boolean exited;
- readonly attribute long exitStatus;
- readonly attribute boolean signaled;
- readonly attribute string terminatingSignal;
- readonly attribute boolean coreDump;
- readonly attribute boolean aborted;
- readonly attribute string reason;
- };

- # of CPUs attribute

Job States

UNDETERMINED,
QUEUED_ACTIVE,
SYSTEM_ON_HOLD,
USER_ON_HOLD,
USER_SYSTEM_ON_HOLD,
RUNNING,
SYSTEM_SUSPENDED,
USER_SUSPENDED,
USER_SYSTEM_SUSPENDED,
DONE,
FAILED



Back-up

DRMAA Placement

- On top of DRM systems
- On top of Globus
- Beneath GRAM
- UNICORE TSI interface to DRMSs
- CoG adapter
- On top of CoG
- Interfaced by a Portal, application, shell
- Portable command line utilities (qsub, qstat)

Job Template

- Functions to create/delete job template
  - job_template *drmaa_allocate_job_template (void)
  - void drmaa_delete_job_template (job_template *jt)
- Setter/getter job template routines
  - int drmaa_set_attribute(job_template *jt, char *name, char *value);
  - int drmaa_set_vector_attribute(job_template *jt, char *name, char **values);
  - char* drmaa_get_attribute(job_template *jt, char *name);
  - char** drmaa_get_vector_attribute(job_template *jt, char *name);