# Sun Grid Engine Resource Broker Adaptor for JavaGAT

Ole Christian Weidner

August 18, 2006

# Contents

# 1 SGE and the DRMAA API

## 1.1 Introduction to DRMAA

DRMAA or Distributed Resource Management Application API is a high-level API specification for the submission and control of jobs on Distributed Resource Management (DRM) systems. It allows an application programmer to access Grid resources through a single API without having to care for the resource manager waiting on the other end. Thus DRMAA has the same functionality as the JavaGAT resource management component - it even uses a similar runtime adaptor loading concept.

## 1.2 SGE Remote Execution Capabilities

# 2 DRMAA to GAT API Mapping

## 2.1 Job States

Mapping job states from DRMAA to SGE is a little weird because of the limited number of states provided by GAT. The following table shows the current mapping. Note that DRMAA `*_SUSPENDED` and `DONE` states are both mapped to the GAT `STOPPED` state, so you can't check for job completitition just by querying the GAT job state when using the SGE adaptor. Furthermore DRMAA doesn't provide states for `PRE_STAGING` and `POST_STAGING`.

| GAT Job States | | SGE DRMAA Job States | |
|---|---|---|---|
| 00 | INITIAL | -- | -- |
| 01 | SCHEDULED | 16 | QUEUED_ACTIVE |
| 02 | RUNNING | 32 | RUNNING |
| 03 | STOPPED | 33 | SYSTEM_SUSPENDED |
| | | 34 | USER_SUSPENDED |
| | | 35 | USER_SYSTEM_SUSPENDED |
| | | 48 | DONE |
| 04 | SUBMISSION_ERROR | 64 | FAILED |
| 05 | ON_HOLD | 16 | SYSTEM_ON_HOLD |
| | | 18 | USER_ON_HOLD |
| | | 19 | USER_SYSTEM_ON_HOLD |
| 07 | PRE_STAGING | -- | -- |
| 08 | POST_STAGING | -- | -- |
| 08 | UNKNOWN | 00 | UNDETERMINED |

Within GAT you have the possibility to access the GAT job state as well as SGE's DRMAA job state. Normally you shouldn't query the DRMAA state within your grid application because doing this makes your application resource broker dependent - but if it comes to the crunch:

```
01 JobInfo info = Job.getInfo();
02 String sge_state = info.get("resManState");
```

This piece of code gets the DRMAA state number (see table) from the JobInfo HashMap (as String). Note that you can't assume that all adaptor implementations provide a resManState field in the JobInfo map. However, every adaptor provides a `getState()` methode to retrieve the GAT state. Use the following code to get the job's state (as Integer):

```
JobInfo info = Job.getInfo();
int sge_state = info.getState();
```

As mendtioned above, you can't assume that a job is done just because it's in GAT `STOPPED` state. It could also be just suspended for some reason. The proper way to check for job completition is to check for the job's state and if it equals `STOPPED` check for the job's exit status. If it's defined, the job is actually done - if not, something went wrong:

```
int state = job.getState();

if (state.equals("STOPPED")) {
    if ( job.getExitStatus() == -255 )
        // Something went wrong - -255 means undefined
    else
        // We have a return code - job is done!
        System.out.println("Job exited with: "job.getExitStatus();
}
```

## 2.2 Job Control

Job control in DRMAA is handled via the `Session.control(java.lang.String jobID, int action)` method, where `action` is an integer macro defining the desired action. The following table shows the current mapping between the DRMAA job actions and the GAT job control interface defined in `org.gridlab.gat.resources.Job`:

| GAT Job Methods | SGE DRMAA Job Controls | |
|---|---|---|
| `Job.unSchedule()` | 4 | `TERMINATE` |
| `Job.stop()` | 4 | `TERMINATE` |
| `Job.checkpoint()` | -- | -- |
| `Job.migrate()` | -- | -- |
| `Job.cloneJob()` | -- | -- |

Note that the job control implementation is not yet completed. Although DRMAA does not directly support actions for checkpointing, migrating and cloning, it should be not very difficult to combine some of the DRMAA actions to emulate the particular functions.

## 2.3 Job Exit Value

With the JavaGAT 1.5 Job API extension there's now a function to query the job's exit value. Job exit value means the return code of the executable submitted via Java-GAT. For example the exit value of `/bin/false` is `1` and the exit value of `/bin/true` is `0`.

The job's exit value is retreived by using the DRMAA `getExitStatus()` method. However, this function only returns a value if the job has exited. If you trigger the `Job.getExitStatus()` function in JavaGAT while the job is still running, you'll get a `-255` as return value. You'll also get a `-255` when the job has exited but never returned an exit value.

## 2.4 Job Info

The job info Map contains the entries listed in the following table. The GAT API specification defines `hostname`, `scheduletime`, `starttime`, `stoptime` and `checkpointable` as supported entries - however, the SGE broker adaptor extends this list with some (hopefully) usefull entries.

| Name | Description |
|---|---|
| `hostname` | Name of the host on which the job is running. |
| `scheduletime` | Time indicating when the job was scheduled.<br>NOT IMPLEMENTED (will always return `null`). |
| `starttime` | Time indicating when the job was started. |
| `stoptime` | Time indicating when the job was stopped. |
| `checkpointable` | Indicating if the job is checkpointable.<br>NOT IMPLEMENTED (will always return `0`). |
| `resManName` | Returns the name of the resource manager.<br>Will always return "Sun Grid Engine". |
| `resManState` | Returns the DRMAA job state. |
| `jobID` | Returns the job's ID assigned by SGE |

# 3 Using the Adaptor

## 3.1 Dependencies

## 3.2 Build and Run your Application