NAREGI GridVM
Basic Execution Service API

GGF13 BES BOF
NAREGI Project, Japan
NAREGI WP1 middleware is a functioning prototype implementation of OGSA-EMS from advanced to basic services. The services are mapped to OGSA-EMS V1 architecture as shown below. Basic Execution Service is provided by GridVM.
What is GridVM?

✓ Virtual execution environment on each site
  • Job execution services
  • Resource management services
  • Secure and isolated environment
Requirements for execution service

✓ Provide platform independence
  • Unified interface for heterogeneous platforms and DRMs (Distributed Resource Manager)

✓ Use standards for interoperability
  • JSDL based job description

✓ Support agreement based job execution
  • Resource reservation
  • Provide resource and service based on site’s policy

✓ Enable the execution of meta-computing job
  • Essential for large scale multi-physics jobs that run across heterogeneous platforms
API Overview

✓ Provides execution services as an abstracted DRM.
  • Tried defining services and operations that can be easily mapped to the underlying DRMs
  • Defined an unified Java I/F, which will be migrated to WSRF-based I/F without difficulties.

✓ The following services are provided through methods of GridVMJobService.
  • BES related functionality
    – Job submission/ control/ state query
    – Event notification

<table>
<thead>
<tr>
<th>GridVMJobServiceFactory</th>
</tr>
</thead>
<tbody>
<tr>
<td>+ createJobService()</td>
</tr>
</tbody>
</table>

• Advanced functionality
  • Advance reservation
  • Evaluating local policy at reservation and job submission
  • Job control of Checkpoint, hold and restart
Job Submit/ Control/ Query

✓ “submitJob” method
  • Submits a job with “JSDL”

✓ “controlJob” method
  • Controls a job specified by “SubjobID” according to specified “Action”
  • Action is one of:
    - Suspend, Resume, Delete

✓ “queryJob” method
  • Queries state of a job specified by “SubjobID”
  • Returned state is one of:
    - Reserved, Queued, Running, Suspended
Event Notification

✓ “addGridVMEventListener” method
  • Registers an event listener for event notification
  • Notified event is one of:
    - Queued, Started, Terminated, Suspended, Resumed

```java
//The interface that event listener should implement
public interface GridVMJobEventListener{
    public void actionPerformed(GridVMJobEvent e);
}
```

✓ GridVMJobEvent object
  • Is passed to registered event listeners when an event occurs
  • Contains information on occurred event
GridVMException

✓ GridVMJobService throws an exception for an error

<table>
<thead>
<tr>
<th>Exception Class</th>
<th>Error Code</th>
</tr>
</thead>
<tbody>
<tr>
<td>GridVMInvalidArgumentException</td>
<td>GRIDVM_INVALID_XML</td>
</tr>
<tr>
<td></td>
<td>GRIDVM_UNDEFINED_CONTROL</td>
</tr>
<tr>
<td>GridVMResourceLimitException</td>
<td>GRIDVM_NODE_LIMIT</td>
</tr>
<tr>
<td></td>
<td>GRIDVM_CHECKPOINTING_UNSUCCESSFUL</td>
</tr>
<tr>
<td></td>
<td>GRIDVM_RESERVED_NODE_LIMIT</td>
</tr>
<tr>
<td></td>
<td>GRIDVM_RESERVED_WALLTIME_LIMIT</td>
</tr>
<tr>
<td></td>
<td>GRIDVM_UNKNOWN_QUEUE</td>
</tr>
<tr>
<td></td>
<td>GRIDVM_CPUCOUNT_LIMIT</td>
</tr>
<tr>
<td></td>
<td>GRIDVM_PHYSICALMEMORY_LIMIT</td>
</tr>
<tr>
<td></td>
<td>GRIDVM_VIRTUALMEMORY_LIMIT</td>
</tr>
<tr>
<td></td>
<td>GRIDVM_CPUCOUNT_LIMIT</td>
</tr>
<tr>
<td>GridVMUnauthorizedAccessException</td>
<td>GRIDVM_UNAUTHORIZED_USER</td>
</tr>
<tr>
<td>GridVMUnexpectedControlException</td>
<td>GRIDVM_INVALID_CONTROL</td>
</tr>
</tbody>
</table>

and more ...
public interface GridVMJobService{

    // Control Action
    public static final int ACTION_SUSPEND;
    public static final int ACTION_RESUME;
    public static final int ACTION_HOLD;
    public static final int ACTION_RELEASE;
    public static final int ACTION_DELETE;
    public static final int ACTION_CHECKPOINT;

    // Job Status
    public static final int JOBQUERY_RESERVED;
    public static final int JOBQUERY_HELD;
    public static final int JOBQUERY_QUEUED;
    public static final int JOBQUERY_RUNNING;
    public static final int JOBQUERY_SUSPENDED;

    // Reservation
    public org.w3c.dom.Document makeReservation(org.w3c.dom.Document jsdl)
    throws GridVMException;
    public void cancelReservation(String subJobID) throws GridVMException;
    public org.w3c.dom.Document queryReservation(String subJobID) throws GridVMException;

    // Job control
    public void submitJob(org.w3c.dom.Document jsdl) throws GridVMException;
    public void controlJob(String subJobID, int action,
        org.w3c.dom.Document information) throws GridVMException;
    public int queryJob(String subJobID) throws GridVMException;

    // Event Listener
    public void addGridVMJobEventListener(GridVMJodEventListener listener);
}
Discussions about BES draft

Based on “Basic EMS service” and “A Basic Execution Service V0.3”

✓ GridVM seems to have no major inconsistency with current BES idea.

✓ From our perspective, the following might be useful as BES functionality.
  
  • Event Notification
    - Efficient compared with polling of job state.
    - Need not keep track of a terminated job after its event of termination is notified.
  
  • Exit Status of a job
    - Need to know the exact reason of job fail.
    - Necessary for controlling a complex workflow by advanced EMS.