Lesson 7: Portlet for special job submission

Mario Torrisi – University of Catania - Italy (mario.torrisi@ct.infn.it)
Sci-GaIA Winter School
Outline

- Manage Special Jobs on DCIs
  - A brief recap on the G&C Engine seen in the last section
  - Special Jobs
    - Job Collection
    - Parametric Job
    - Workflow 1-N-1
  - Grid Engine API to execute Special Job on DCIs

- Portlet template
  - Get code from github repository
  - Compile and test

- Customize

- Summary and conclusion
Manage special jobs on DCIs
Grid & Cloud Engine

- Allows to develop applications able to submit jobs towards DCI in a easy way
- Exposes a set of intuitive APIs
- The developer has only to submit the job, the Job Engine:
  - periodically checks the job status
  - automatically downloads and (if set) sends an email to notify the user
Job Engine
(Features)

- Middleware independent
  - Based on JSAGA, an implementation of the SAGA standard
- Scalability
  - Is able to manage many parallel job submissions requests
- Performance
  - Hides all DCIs interactions delays to the final user
  - Is able to submit thousands of jobs in a short time according to the hardware capabilities
- Accounting
  - Is fully compliant with the **EGIVO Portal Policy** and **EGI Grid Security Traceability and Logging Policy**
- Fault tolerance
  - Provides mechanisms to guarantee job submission that hide infrastructure failures.
- Workflow
  - Provides easy mechanisms to natively support parallel applications
    - **JobCollection**
    - **Parametric Job**
    - **Workflow 1-N-1**
Job Collection

- **Job Collection**: is a set of independent jobs that run in parallel.
- When all jobs are successfully completed the whole collection becomes **DONE**.
Parametric Job

- **Parametric Job**: is a set of jobs that run in parallel having the same executable, but with different arguments (e.g., input parameters)
- When all jobs are successfully completed the whole parametric job becomes **DONE**
Workflow 1-N-1

- **Workflow 1-N-1** is a Special Job consisting of two different levels of jobs:
  - First level jobs is a set of independent jobs that run in parallel
  - Final (Collector) job that collects all first level jobs outputs and uses them as input

- When all first level jobs are successfully completed the final job will be submitted and when it becomes **DONE** the whole workflow can be considered completed
Manage Special Jobs

- Parametric Job and Workflows 1-N-1 are designed as generalisation of Job Collection
JobCollection.java

- This class handles the behaviour of the **Job Collection**
- It is responsible for:
  - creation
  - status update
  - close the collection when all jobs end
**JobParametric.java**

- This class handles the behaviour of the **Parametric Job**
- Inherits all attributes and methods defined in its parent class
- Adds a new attribute that specify the single executable
WorkflowN1.java

- This class handles the behaviour of the Workflow 1-N-1
- Inherits all attributes and methods defined in its parent class
- Adds 3 new attributes:
  ```java
  private Long idFinalJob;
  private GEJobDescription finalJobDescription;
  private String[] inputFilePrefixes;
  ```
- Overrides some JobCollection methods
Example

1. Define Infrastructures

InfrastructureInfo infrastructures[] = new InfrastructureInfo[2];
String wmsList[]={"wms://wmsdecide.dir.garr.it:7443/glite_wms_wmproxy_server"};
String wmsList2[]={"wms://wms.magrid.ma:7443/glite_wms_wmproxy_server"};

infrastructures[0] = new InfrastructureInfo("gridit",
   "ldap://gridit-bdii-01.cnaf.infn.it:2170", wmsList,
   "etokenserver.ct.infn.it", "8082", "332576f78a4fe70a52048043e90cd11f",
   "gridit", "gridit");

wmsList2,
   "etokenserver.ct.infn.it", "8082", "bc681e2bd4c3ace2a4c54907ea0c379b",
   "eumed", "eumed");
Example

2. Define jobs descriptions

```java
ArrayList<GEJobDescription> descriptions = new ArrayList<GEJobDescription>();
for (int i = 0; i < 2; i++) {
    GEJobDescription description = new GEJobDescription();
    description.setExecutable("/bin/sh");
    switch (i) {
        case 0:
            description.setArguments("hostname.sh");
            description.setInputFiles("./hostname.sh");
            break;
        case 1:
            description.setArguments("ls.sh");
            description.setInputFiles("./ls.sh");
            break;
    }
    description.setOutputPath("/tmp");
    description.setOutput("myOutput-" + i + ".txt");
    description.setError("myError-" + i + ".txt");
    descriptions.add(description);
}
```
Example
(Job Collection)

3. Create JobCollection object

```java
JobCollection collection = new JobCollection("mtorrisi", "Collection - TEST - ", "/tmp", descriptions);
```

4. Submit JobCollection

```java
JobCollectionSubmission tmpJobCollectionSubmission = new
JobCollectionSubmission("jdbc:mysql://localhost/userstracking", "tracking_user", "usertracking", collection);

tmpJobCollectionSubmission.submitJobCollection(infrastructures, "193.206.208.183:8162", 10);
```
Example (Parametric Jobs)

3. Create JobParametric object

```java
JobCollection p = new JobParametric("test",
    "Parametric Job - mtorrisi", "mario.torrisi@ct.infn.it",
    "/tmp", descriptions, "hostname.sh");
```

4. Submit Parametric Job

```java
JobCollectionSubmission tmpJobCollectionSubmission = new JobCollectionSubmission(
    "jdbc:mysql://localhost/userstracking", "tracking_user",
    "usertracking", p);
    tmpJobCollectionSubmission.submitJobCollection(infrastructures,
    "193.206.208.183:8162", 10);
```

*The first two steps are common to the "Job Collection" submission
Example (Workflow 1-N-1)

3. Create Final Job Description

GEJobDescription finalJobDescription = new GEJobDescription();
finalJobDescription.setExecutable("ls.sh");
finalJobDescription.setArguments("./ls.sh");
String tmp = "";
for (int i = 0; i < descriptions.size(); i++) {
    if (tmp.equals(""))
        tmp = descriptions.get(i).getOutput();
    else
        tmp += "," + descriptions.get(i).getOutput();
}
finalJobDescription.setInputFiles(tmp + ",./ls.sh");
finalJobDescription.setOutputPath("/tmp");
finalJobDescription.setOutput("myOutput-FinalJob.txt");
finalJobDescription.setError("myError-FinalJob.txt");

*The first two steps are common to the "Job Collection" submission
Example
(Workflow 1-N-1)

4. Create WorkflowN1 object

JobCollection wf = new WorkflowN1("mtorrisi", "Workflow N-1", "/tmp", descriptions, finalJobDescription);

5. Submit Workflow 1-N-1

JobCollectionSubmission tmpJobCollectionSubmission = new JobCollectionSubmission( "jdbc:mysql://localhost/userstracking", "tracking_user", "usertracking", wf);
    tmpJobCollectionSubmission.submitJobCollection(infrastructures, "193.206.208.183:8162", 10);
Template Special Jobs Portlet
Template Special Jobs Portlet

- The **Template Special Jobs Portlet** is a complete example of portlet able to submit special Job towards distributed environments.
- Its Java code extends the Liferay MVCPortlet class and uses JSP pages to generate the input GUI.
Template Special Jobs Portlet
(Features)

- It provides a example of:
  - Managing input elements from web forms as application input
  - Managing portlet configuration to handle distributed infrastructure settings
- It is a multi-infrastructure multi-middleware portlet to execute special jobs on several infrastructures and middleware (currently supports: gLite-based Grids, HPC Clusters, Cloud resources)
- It can be used as starting point to develop your own application
  - A customization script is provided in order make its reuse easy
Template Special Jobs Portlet (Usage)

1. Select special job type
2. Specify task number
3. Click “OK”
4. Fill input fields
5. Specify a label (optional)
Template Special Jobs Portlet
(Build & Deploy)

- Clone the source repository
  - cd /path/to/plugin-sdk/portlet
  - git clone https://github.com/sci-gaia/template-special-job-portlet
  - cd template-special-job-portlet
  - ant deploy
  - Check server.log to see
  - Add to page

- Remember to insert a new entry in the GridOperations table
Before testing special job execution

- Open the VPN or be sure the eTokenserver allows incoming connections on port 8082 from your portal IP address

- Check the eTokenserver service is reachable
  - From the interface generate the robot proxy request
  - Execute `curl` or `wget` on the generated request
Template Special Jobs Portlet
(Test)
Special Job Execution
(Log inspection)

- Liferay server.log file reports:
  - A full dump of the portlet preference values
    
    ```
    [mi_hostname_portlet:108] dump:
    Infrastructure #1
    enableInfrastructure : 'yes'
    nameInfrastructure : 'EUMEDGRID-Support infrastructure'
    acronymInfrastructure: 'EUMEDGRID'
    ```
  - The GridEngine Initialization
    ```
    INFO JSagaJobSubmission - Getting adaptor name...
    JSagaJobSubmission - Using adaptor: wms
    ```
  - The Robot proxy retrieval
    ```
    INFO RobotProxy - proxyPath=/tmp/7f7e1e98-0fd1-4ebb-a1ae-0627efddf600
    INFO RobotProxy - get proxy:
    http://etokenserver.ct.infn.it:8082/eTokenServer/eToken/332576f78a4fe70a52048043e90cd11f?voms=gridit:gridit&proxy-renewal=true
    ```
Special Job Execution
(Log inspection)

- The JSAGA job submission string

```
[submitJob][JSagaJobSubmission:1057]
jobSandbox:/tmp/jobOutput/DemoCollection842016195312_0_20/<myOutput-0.txt,/tmp/jobOutput/DemoCollection842016195312_0_20/<myError-0.txt
[submitJob][JSagaJobSubmission:1057]
jobSandbox:/tmp/jobOutput/DemoCollection842016195312_2_20/<myOutput-2.txt,/tmp/jobOutput/DemoCollection842016195312_2_20/<myError-2.txt INFO
[submitJob][JSagaJobSubmission:1057]
jobSandbox:/tmp/jobOutput/DemoCollection842016195312_1_22/<myOutput-1.txt,/tmp/jobOutput/DemoCollection842016195312_1_22/<myError-1.txt
```

- The input sandbox file transfers

Connecting to Gsiftp service at: wms014.cnaf.infn.it:2811...

- The job id and the job status thread execution

Job Submitted: [wms://wms01.afroditi.hellasgrid.gr:7443/glite_wms_wmproxy_server]-[https://wms01.afroditi.hellasgrid.gr:9000/PxAwp6kRj7iJYq5veuhhTw]
Job Submitted: [wms://wms01.afroditi.hellasgrid.gr:7443/glite_wms_wmproxy_server]-[https://wms01.afroditi.hellasgrid.gr:9000/SmCmWIXr3QLv_O4HsRD3HA]
Job Submitted: [wms://wms01.afroditi.hellasgrid.gr:7443/glite_wms_wmproxy_server]-[https://wms01.afroditi.hellasgrid.gr:9000/kvBVjiDsNokHl9XzJ6Iw9g]
Portlet Code
Portlet Code
Portlet code
(Java Classes)

- **AppInfrastructureInfo.java**
  - The same class of the Template portlet

- **AppInput.java**
  - Few changes to support the input required by the special jobs

- **AppPreferences.java**
  - The same class of the Template portlet

- **ConfigurationActionImpl.java**
  - The same class of the Template portlet

- **TemplateSpecialJobPortlet.java**
  - We will see this class more in depth later …
Portlet code
(Java Classes)

- Constats.java
  - The same class of the Template portlet

- Utils.java
  - The same class of the Template portlet
Portlet code

(TemplateSpecialJobPortlet.java)
private void submitJobCollection(AppPreferences preferences,
   AppInput appInput, InfrastructureInfo[] enabledInfrastructures) {

   ArrayList<GEJobDescription> descriptions = new ArrayList<GEJobDescription>();

   for (int i = 0; i < appInput.getTaskNumber(); i++) {
      GEJobDescription description = new GEJobDescription();
      switch (CollectionType.valueOf(appInput.getCollectionType())) {
         case JOB_COLLECTION:
         case WORKFLOW_N1:
            // Add code for JOB_COLLECTION and WORKFLOW_N1
            break;
         case JOB_PARAMETRIC:
            // Add code for JOB_PARAMETRIC
            break;
      }
   }
}
Portlet code
(TemplateSpecialJobPortlet.java)

case JOB_COLLECTION:
collection = new JobCollection(appInput.getUsername(),
    appInput.getJobLabel(), "/tmp", descriptions);
    break;

case WORKFLOW_N1:
GEJobDescription finalJobDescription = new GEJobDescription();
...

    finalJobDescription.setInputFiles(tmp);
    finalJobDescription.setOutputPath("/tmp");
    finalJobDescription.setOutput("myOutput-FinalJob.txt");
    finalJobDescription.setError("myError-FinalJob.txt");
collection = new WorkflowN1(appInput.getUsername(),
    appInput.getJobLabel(), "tmp", descriptions,
    finalJobDescription);
    break;

case JOB_PARAMETRIC:
collection = new JobParametric(appInput.getUsername(),
    appInput.getJobLabel(), "/tmp", descriptions,
    appInput.getExecutable());
    break;
// GridEngine' JobCollectionSubmission job submission object
JobCollectionSubmission tmpJobCollectionSubmission = null;

...

tmpJobCollectionSubmission = new
  JobCollectionSubmission(collection);

...

// Ready now to submit the Job
tmpJobCollectionSubmission.submitJobCollection(enabledInfrastructures, portalIPAddress, applicationId);
Customization script

customize.sh
References

- Portlet source code repository
  - [https://github.com/sci-gaia/template-special-job-portlet](https://github.com/sci-gaia/template-special-job-portlet)

- Discussion Forum

- EGI IGTF Release

- VOMS Clients guide
Thank you!
mario.torrisi@ct.infn.it
sci-gaia.eu