Lessons Learned from UNICORE EMI-ES Adoption towards Improved Open Standards

Shahbaz Memon et al.
Jülich Supercomputing Center
29-03-2012
Outline

• Motivation
• EMI-ES Introduction
• UNICORE Overview
• UNICORE EMI-ES Implementation
• OGF Standards
• EMI-ES enhancements
• Conclusion
Motivation

• EMI mandate to support diversified scientific communities
• Proprietary interfaces was not useful for e-Infrastructures
• Job management standards permeated the Grid middleware stacks to attract myriad set of scientific user communities
• Evolving application requirements influence middleware stacks – EMI user requirements
• They can be improved with advanced execution service concepts
• EMI-ES: a step towards the next generation of Job management standards.
Approach

EMI ES Client(s)

ARC CE  gLite Cream  UNICORE

Backend systems

03/04/2012  EGI Community Forum 2012
EMI-ES Overview

• Web Service interface definitions
  – Create and manage vector of activities
  – Expose capabilities as GLUE-2 instance
  – Delegation (issue a proxy cert for data staging)

• Job information Model - ADL
  – Describes a job to be executed by EMI-ES
  – Encompass execution environments
  – Supporting the concept of serial and parallel jobs
EMI-ES Component Architecture

<<Component>>
Activity-Factor
- <<interface>> Create
- <<interface>> ResourceInfo
- <<interface>> Delegation

<<Component>>
Activity-Manager
- <<interface>> ActivityManagement
- <<interface>> ActivityInfo
- <<interface>> Delegation
EMI-ES Job Description: ADL

Activity Description Language

**Activity Identification**
- Name
- Description
- Annotation

**Data Staging**
- Input
- Output
- Source
- Target
- DelegationID

**Resources**
- CPU Time
- Benchmark
- Network
- Slot
- RTE
- PE
- QN

**Application**
- Exec
- Env
- Args
- IO Redirect
- Pre/Post Exec...
- RL
Activity State Model – Optimal Flow

- State model is exposed to clients (not necessarily used by the implementation internally)
- Each state may be assigned multiple attributes
- Validating, Server-Paused, Provisioning, App-Failure,..
UNICORE Overview

- Integrated, complete Grid middleware stack including Graphical & command line clients
- Focus on ease of use (both end users and admins)
- Lightweight and platform independent, coded in Java and Perl
- Supports many resource management systems (PBS, Torque, LSF) and operating systems (Linux/Unix, Mac OS X, Windows)
- Strong support for applications and workflows
1. Submit job

2. Remote request received by EMI-ES (alternatively OGSA-BES, UAS)

3. XNJS generates a TSI script from the job description and local configuration stored in IDB file

4. Send TSI Script

5. Submit TSI script ("llsubmit")

Cluster login node

Target System Interface

Local Batch System
OGSA-BES and JSDL

• OGSA-BES: a web service interface to manage and monitor Grid jobs
  – Create single activity
  – Get activities statuses
  – Terminate activities

• JSDL
  – Job request model
  – Job identification, Resources, Application, and Data staging elements
OGF Profiles

- Scope the use of multiple standards for a particular use case
- In OGSA-BES and JSDL
  - HPC-BP, HPC-FSP
  - JSDL-SPMD, JSDL-HPC
OGF JSDL Extensions

• Application
  – Pre & Post-Executable
  – RemoteLogging
  – WipeTime
  – Notification

• Resources
  – Runtime Environment
  – Parallel Environment
GLUE2 Extensions

- ComputingActivity
  - StageInDirectory
  - StageOutDirectory
  - SessionDirectory
  - ComputingActivityHistory
  - ComputingActivityProgress
# EMI-ES and OGSA-BES

<table>
<thead>
<tr>
<th>Approach</th>
<th>EMI-ES</th>
<th>OGSA-BES</th>
</tr>
</thead>
<tbody>
<tr>
<td>Manage Activities</td>
<td>CreateActivities</td>
<td>CreateActivity (Single)</td>
</tr>
<tr>
<td></td>
<td>Pause</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Resume</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Cancel</td>
<td>TerminateActivities (Vector)</td>
</tr>
<tr>
<td></td>
<td>Notify</td>
<td>Use of WSN</td>
</tr>
<tr>
<td></td>
<td>Wipe</td>
<td></td>
</tr>
<tr>
<td>Monitor</td>
<td>ListActivities (return on Ids)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>GetActivityStatus</td>
<td>GetActivityStatuses</td>
</tr>
<tr>
<td></td>
<td>GetActivityInfo</td>
<td>GetActivityDocuments</td>
</tr>
</tbody>
</table>

03/04/2012  
EGI Community Forum 2012
EMI-ES and OGSA-BES

<table>
<thead>
<tr>
<th>Approach</th>
<th>EMI-ES</th>
<th>OGSA-BES</th>
</tr>
</thead>
<tbody>
<tr>
<td>Monitor Computing Service</td>
<td>GetResourceInfo</td>
<td>GetFactoryAttributes</td>
</tr>
<tr>
<td></td>
<td>QueryResourceInfo</td>
<td>Not in the spec (only WSRF renderings)</td>
</tr>
<tr>
<td></td>
<td>GetActivityInfo</td>
<td>GetActivityDocuments</td>
</tr>
<tr>
<td>Information Model</td>
<td>ResourceInfo (Glue2: ComputingManager,</td>
<td>FactoryAttributes (BES)</td>
</tr>
<tr>
<td></td>
<td>ComputingEndpoint, ExecutionEnvironment,</td>
<td></td>
</tr>
<tr>
<td></td>
<td>ApplicationEnvironment)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>ActivityInfo (Glue2: ComputeActivity)</td>
<td>ActivityDocument (JSDL)</td>
</tr>
<tr>
<td>Data Staging</td>
<td>Client Initiated</td>
<td>Server Initiated</td>
</tr>
<tr>
<td></td>
<td>Server Initiated</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Stage-in /Session / Stage-out directories</td>
<td>Not distinguished</td>
</tr>
</tbody>
</table>

EMI INFSO-RI-26161
03/04/2012
EGI Community Forum 2012
17
Conclusion

• Comes with suite of interfaces, activity state model, support of vector operations, integrated job description language
• Expose capabilities and state by associating GLUE2 entities
• EMI-ES supported by ARC, gLite, and UNICORE
• Job standards and profiles could be improved to support production infrastructure requirements via EMI-ES extensions