JMEA
Job Manager Enterprise Application

Thomas Soddemann, RZG
Overview

• RZG and DEISA
• DEISA and its resources
• Access to Resources in DEISA (s.a. next talk)
• Material Science and Plasma Physics Portal requirements
• JMEA
History of supercomputing at the RZG

1962: IBM 7090
1969: IBM 360/91
1979: Cray-1
1998: Cray T3E/816
1999: NEC SX-5/3C
2002/2003: IBM p690
User community

- Supercomputing and Application Support
- Data management and long-term archives
- Data acquisition systems for fusion experiments
- Bioinformatics platform

Users from Max-Planck-Institutes all over Germany, Italy, and the Netherlands
DEISA – Distributed European Infrastructure for Supercomputing Applications
DEISA

DEISA is an European Supercomputing Service built on top of existing national services.

DEISA deploys and operates a persistent, production quality, distributed supercomputing environment with continental scope.
DEISA

AIX distributed super-cluster

THE DEISA SUPERCOMPUTING GRID

Vector systems (NEC, …)

Linux systems (SGI, IBM, …)

Unicore Summit 2006, Dresden, Germany
August, 31st
Deisa Network Status

Dedicated network infrastructure using „Premium IP“

Participating NRENs

<table>
<thead>
<tr>
<th>Country</th>
<th>NREN</th>
</tr>
</thead>
<tbody>
<tr>
<td>ES</td>
<td>RedIRIS, Spain</td>
</tr>
<tr>
<td>FI</td>
<td>FUNET, Finland</td>
</tr>
<tr>
<td>FR</td>
<td>RENATER, France</td>
</tr>
<tr>
<td>DE</td>
<td>DFN, Germany</td>
</tr>
<tr>
<td>IT</td>
<td>GARR, Italy</td>
</tr>
<tr>
<td>NL</td>
<td>SURFNET, The Netherlands</td>
</tr>
<tr>
<td>UK</td>
<td>UKERNA/JANET, UK</td>
</tr>
</tbody>
</table>
Unicore Summit 2006, Dresden, Germany

MC-GPFS Multiple Network Streams

- Use of many I/O Server implies high disk performance
- Saturation of a 1 Gbit/s or 10 Gbit/s line is easier to achieve
The DEISA Super Cluster in 2005/2006

AIX IBM domain

- ECMWF (UK)
- IDRIS (FR)
- CSC (FI)
- AIX IBM domain

HPC Common Global File System
- similar architectures / operation systems
- High bandwidth (10 Gbit/s)

CINECA (IT)
- Jülich (DE)

RZG (DE)

HPC Common Global File System
- various architectures / operating systems
- High bandwidth (10 Gbit/s)
- More than 100 TFlop/s, 50 TB memory
Ways of accessing resources: CLI

ssh, qsub/llsubmit, qstat/llq, …
Local Resource Management

- Load Leveler
- LSF
- OpenPBS/PBSpro
- Sun Grid Engine
- Torque

Obvious disadvantages:

- Separate batch script for each environment
- No job rerouting

- MC-LoadLeveler
Ways of accessing resources: Rich Client Solution
Ways of accessing resources: Web Portal Solution
UNICORE deployment in DEISA
UNICORE Configuration (RZG)

- UNICORE Gateway in DMZ
- Remote gateway
- NJS hosted separately

- UNICORE Configuration (RZG)
DEISA Research Activities

JRA1 – Material Sciences
CPMD
CP2K

JRA3 – Plasma Physics
TORB
Requirements for a Portal Solution

- Compute Job Handling
  (submit, cancel, hold, status, …)
  => components holding job information
- Session management
- File staging support
- Remote file system access
- Database access
- User Administration (auth*)

Job Manager
Session Manager
Persistence Manager
Identity Manager
Advantages of Portal Applications

- Give the possibility to hide the complexity of Grid Infrastructures (sensible simplifications vs. mystification MS approach)
- Can give the impression of direct use of an application

**Application Service Provider -> ASP**

- Can be accessed from almost everywhere
Architecture of a Web Based Enterprise Solution
Architecture of a Web Based Enterprise Solution

- Web Browser
- Web Application
- Other EA
- JMEA
- Other Middleware
- UNICORE server side
- UNICORE Client
JobManager Methods

- submit (submitting the job request),
- cancel (canceling a job request which is not being executed)
- delete (delete a finished job request),
- kill (kill a running job request)
- halt (halt a job which is being executed)
- resume (resume a previously halted job).

- Publish information about
  - resources
  - status of jobs
  - fetch console output.

- In addition
  - Support of Proxy Certificates
  - Support of Explicit Trust delegation
The JobManager Interface

Allows different implementations:
- UNICORE (primary target)
- Globus
- ...
Arcon Library disadvantages (Multi User Application)

In order to avoid race conditions in multi threaded applications, one should

• omit static variables
  unless they are used for communication between the threads and their access is synchronized.

• synchronized access must not lead to a performance bottlenecks

• Keep in mind that the application might be clustered
Arcon Library disadvantages (Multi User Application)

Arcon Library defines:

- **outcome_dir** which specifies the directory, where streamed files will be stored
- **buffer_size** which reflects the buffer size for connections
- **always_poll** which tells if request are always asynchronous or not.

The abstract class Connection implements three static variables:

- **keep_open** which defines if the next connection is kept open after use.
- **compression** which tells if the transmission should be compressed or not.
- **encrypt** which defines whether the next retrieved connection should be encrypted communication or not.
Arcon Library disadvantages (Multi User Application)

Further disadvantages

- Proprietary Logging
- Exceptions used for control flow rather than for error handling
- Missing support for ETD (at least in the official release)
New UNICORE JobManager library

- Need for a new job management library implementation for UNICORE
- Arcon partly code reused/refactored
- Reimplementation of problematic parts
- ETD support
- Proxy Certificate Support
- Thread safe
Job Management Enterprise Application - JMEA

Stateless Session EJB
Implements JobManager Interface

Message Driven EJB
Performs asynchronous task
(e.g. long lasting communication with NJS)

JobManager MD Bean

JobManager EJB

JobManager Service MBean

Management Bean
Registry for Jobs
Initiates DB updates

ResourceService MBean

Management Bean
Resource information cache

JobManager Library
Job Submission Sequence

Client

JobManagerEJB

JobManagerMBean

JobManagerMDBean

JobManagerImpl

submit Job

submit job

done

done

make job persistent

register Job

done

submit Unicore Job
Job Status Request Sequence

- Client
- JobManagerEJB
- TimerTask
- JobManagerMBean
- JobManagerMDBean
- JobManagerImpl

status
retrieve status
update
get status
make status persistent
Facts on JMEA

Advantages

• Responding fast to client requests
• Scalable (number of client requests)
  – Implies scalable database and container infrastructure
• Fault tolerant (to some extend)

Disadvantages

• Risk of delivering outdated data
• No support UNICORE alternate file transfer
  The web application has files transferred independently (GPFS, CIFS)
Security Solutions: Proxy Certificate Approach
Security Solutions: Explicit Trust Delegation

Unicore Summit 2006, Dresden, Germany
August, 31st
Possible UNICORE5 improvements

AJO
• Default constructors for all classes (unless it does not make sense)
  – Background: Persistence

ETD
• Use of X500Principals instead of whole X509 certificates as user attribute
• Allowing more “direct“ requests for an ETD agent
  – E.g. for Resource information
Conclusion

- JMEA is an EA which proves to work with UNICORE5 in DEISA
- It has all basic features implemented which are needed for successful job management
- It is designed to work with the JRA1/JRA3 Web application
- It can be used in a different context (OMII/GridSAM)
- But, it does not provide a standards based interface

- WS-GRAM (4.0 and 4.1) support is being developed
- UNICORE6 support is hopefully given with WS-GRAM 4.2 support
Thanks

Contributors to the JRA1/JRA3 endeavors:

Johannes Reetz, RZG, Garching, Germany
Daniel Frank, RZG, Garching, Germany

Discussion Partners (GridSAM)

Researchers/Beta-Testers
Matthias Krack, ETH, Zürich, Switzerland
Peter Coveney, UCL, London, England