The DESHL Client

Distributed European Infrastructure for Supercomputing Applications

Malcolm Illingworth, EPCC

DEISA Training Session
Paris, 4th July 2006

What is the DESHL ?

- ***DESHL: DEISA Services for the Heterogeneous management Layer***

- Command line application and application programming interface
- Distributed resource management for UNICORE-based grids
- Submit and manage batch jobs
- Uniform data management across different platforms
- Based on emerging grid standards

JRA7 Objectives

Distributed European Infrastructure for Supercomputing Applications

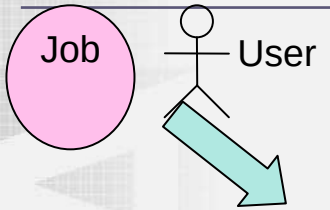
DESHL produced by the DEISA JRA7 Research Activity:

"To develop a single way of coordinating and integrating Open Grid Service Architecture services for distributed resource management in a heterogeneous environment, and to use this to integrate a variety of existing user-level tools to provide the necessary high-level services in:

- authentication, authorisation and accounting;
- job preparation, submission and monitoring;
- data movement for job input and output;
- other areas to be determined by DEISA user requirements."

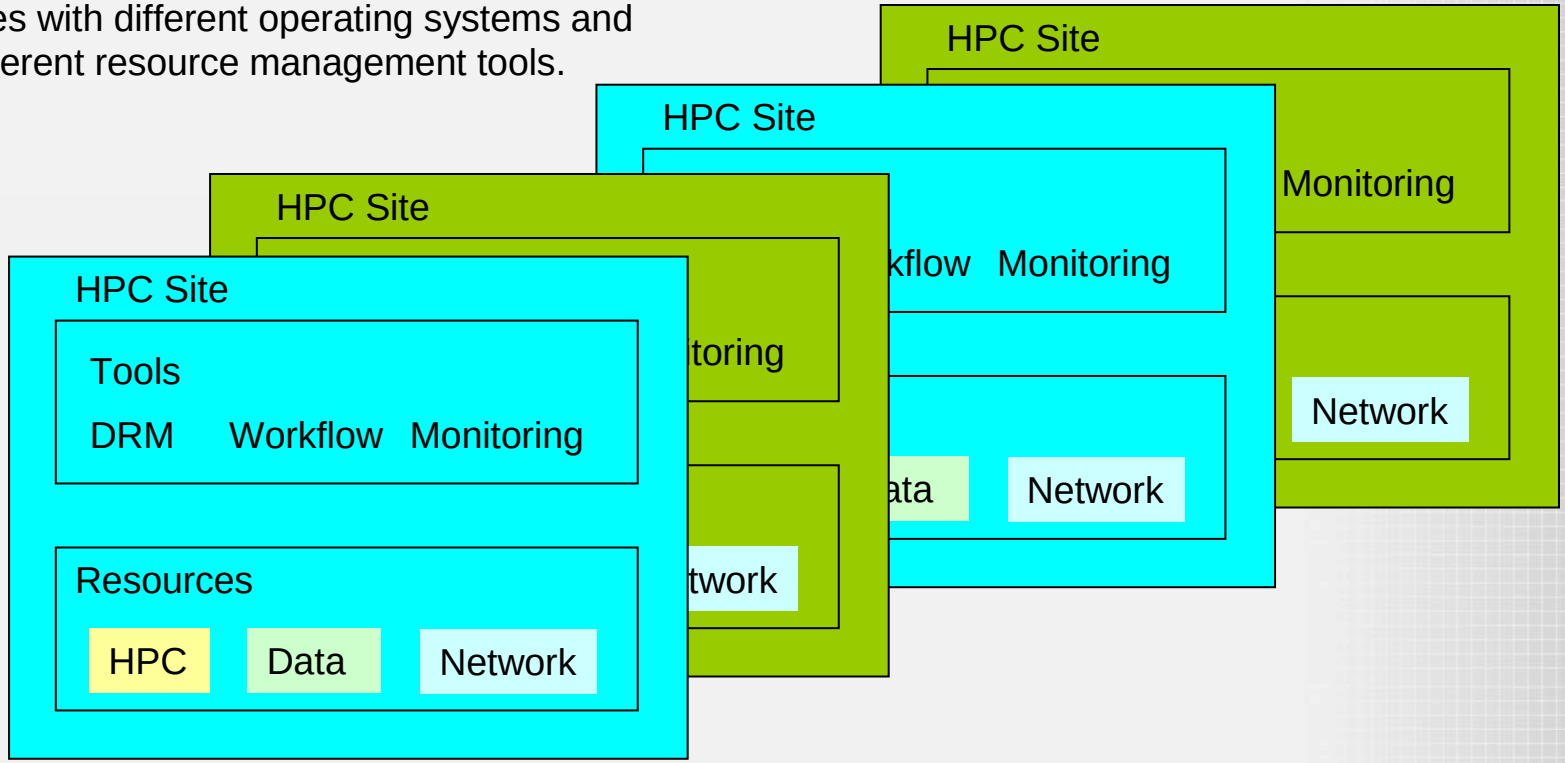
The Big Picture (1)

Distributed European Infrastructure for Supercomputing Applications



User has a compute job or a network or resource monitoring request,…

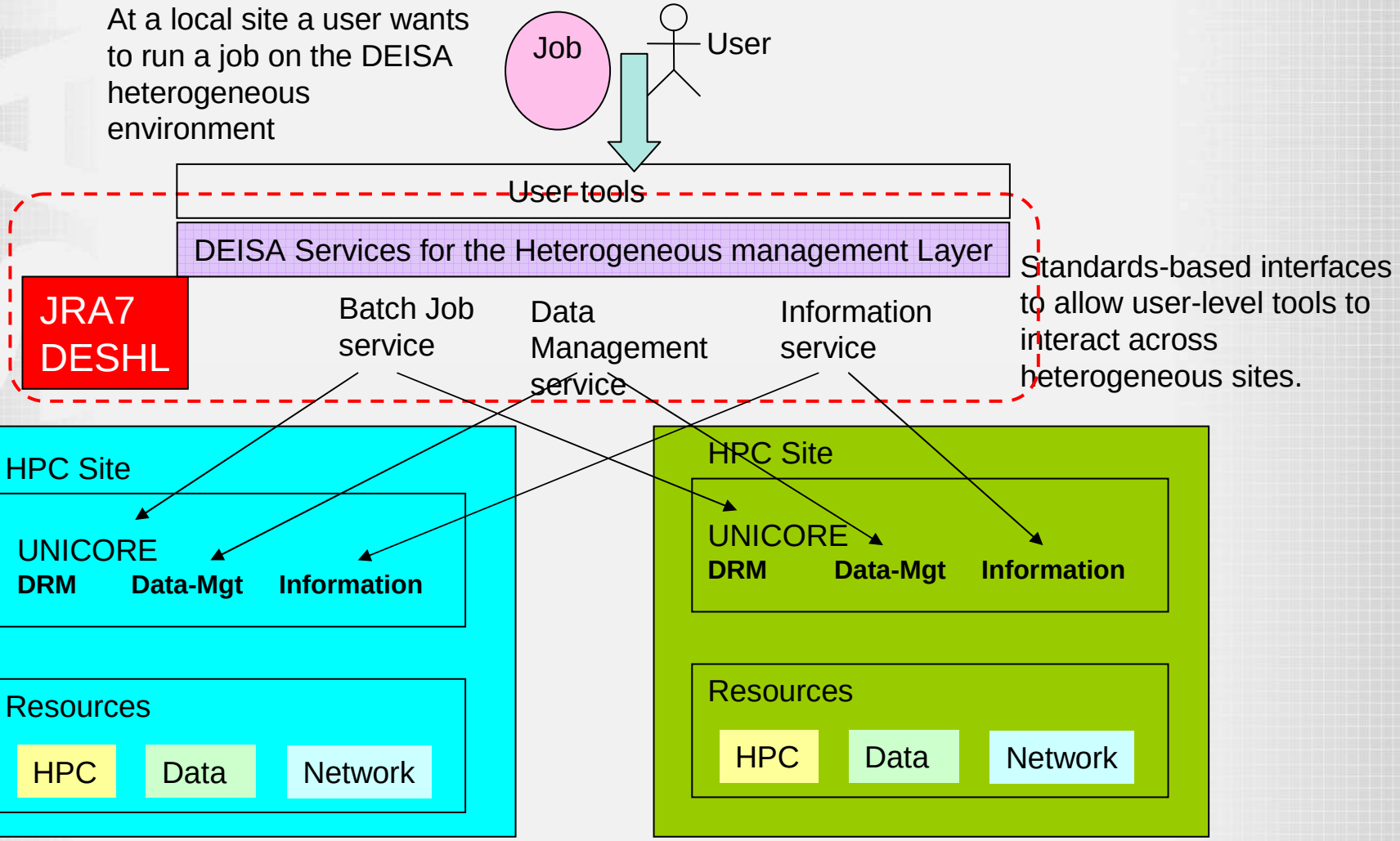
DEISA Heterogeneous Environment: HPC sites with different operating systems and different resource management tools.



The Big Picture (2)

Distributed European Infrastructure for Supercomputing Applications

At a local site a user wants to run a job on the DEISA heterogeneous environment



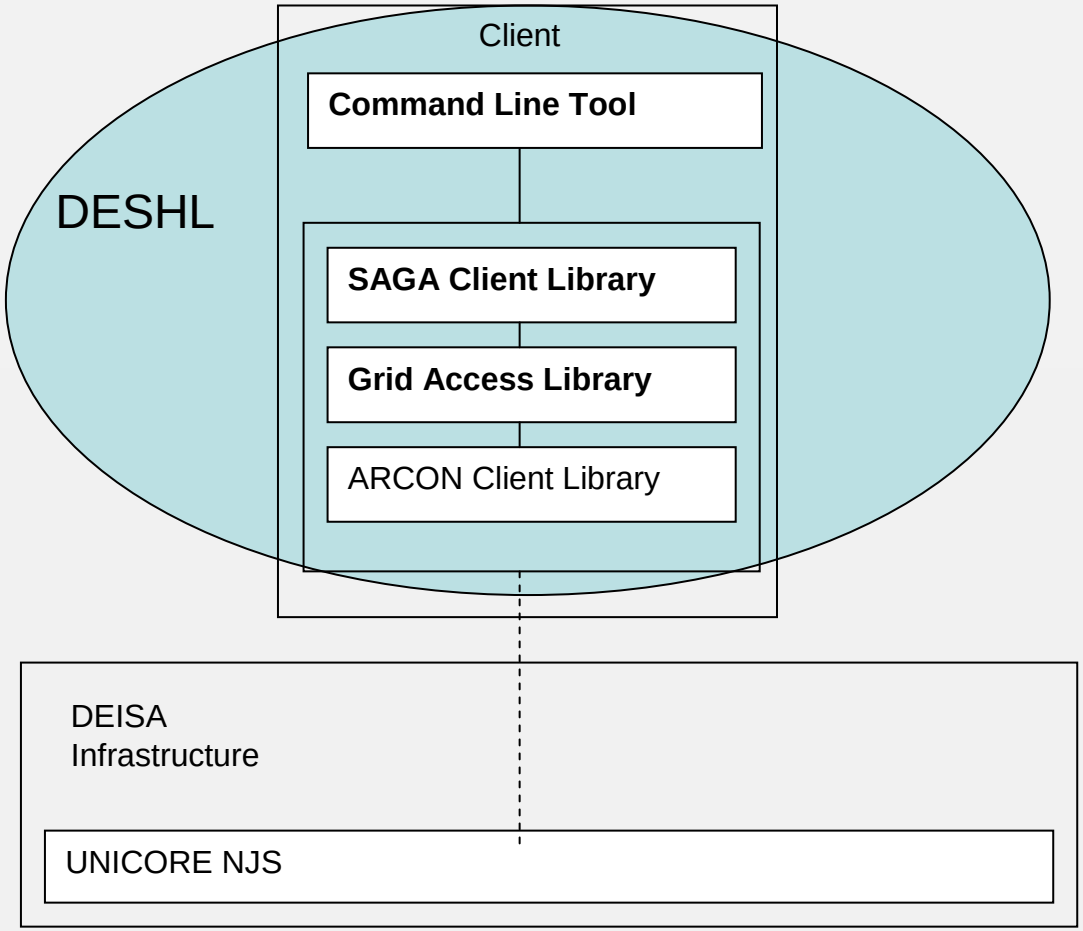
DESHL Client Overview

Distributed European Infrastructure for Supercomputing Applications

- Supplied as a Java command-line application
- Follows Open Grid Service Architecture (OGSA) specification for job batch submission and management
- Uses SAGA (Simple API For Grid Applications) directives for job specification
- Where appropriate, DESHL commands follow the Open Group Technical Standard for Batch Environment Systems
- Layered design to protect against future changes in underlying infrastructure
- Sits on top of existing UNICORE infrastructure

DESHL v3.0 Architecture

Layered Design sitting on top of existing Unicore infrastructure



DESHL Components

Distributed European Infrastructure for Supercomputing Applications

- SAGA Client Library
  - Follows SAGA standards (Simple API For Grid Applications)
  - Hides implementation details of underlying infrastructure
  - Exposes simple, compact API for job management and data staging
  - API can be used to develop simple standards-based grid applications
- Grid Access Library (roctopus)
  - Provides access to UNICORE through ARCON client library, but hides low-level nature of ARCON
  - Provides rich object-oriented interfaces for grid access
  - API can be used to build complex grid applications

DESHL Functionality - Overview

Distributed European Infrastructure for Supercomputing Applications

- Data Transfer
  - upload a file to a DEISA site
  - download a file from a DEISA site
  - delete a file from a DEISA site
  - copy/move a file or directory between DEISA sites
  - list a file or directory on a DEISA site
- Job Management
  - submit a batch job to a DEISA site
  - view the status of a batch job on a DEISA site
  - terminate a batch job at a DEISA site
  - Retrieve stdout and stderr for a completed/terminated job
- Authentication/Authorization is by existing UNICORE mechanisms

  valid X.509 certificates for UNICORE are also valid certificates for DESHL

Client install and configuration

Distributed European Infrastructure for Supercomputing Applications

- Installation is via a GUI configuration tool
- Access to DEISA sites controlled through local configuration file
- Only those DEISA sites which have entries in the host configuration file can be accessed by the DESHL client
- Does not require UNICORE graphical client to be installed or present

Example configuration file

Site Name	Certificate	Password	Alias
ssl://admin.hpcx.ac.uk:4433/EPCC%20HPCx	c:/cer1.p12		hpcx
ssl://admin.hpcx.ac.uk:4433/IDRIS%20ZAHIR	c:/cer2.p12		idris
ssl://admin.hpcx.ac.uk:4433/FZJ%20JUMP	c:/cer2.p12		fzj
ssl://admin.hpcx.ac.uk:4433/RZG%20SP4	c:/cer3.p12		Rzg
#ssl://deisa1.epcc.ed.ac.uk:4010/TEST	c:/cer4.p12		test

- The DESHL client can support individual user certificates for each site, however for DEISA one user certificate is used for all sites.
- Users can define “aliases” for DEISA sites
- Individual sites can be commented out as required

Data Staging

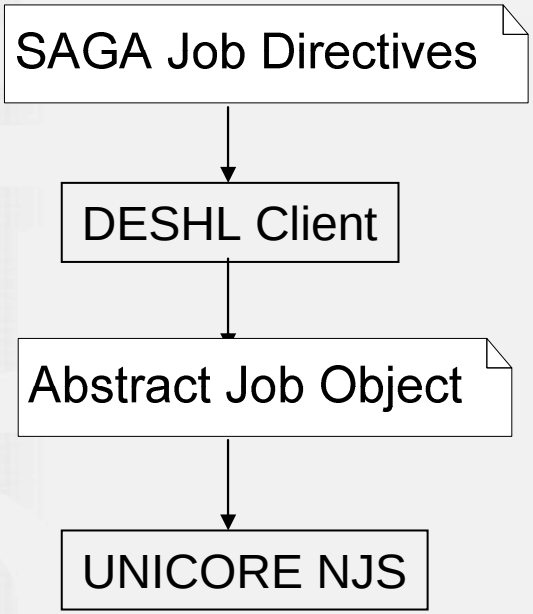
- Uses Unicore “storage” concept - equivalent to a remote mounted filesystem
- Storages for a user configured at the site
- Data staging must be between configured storages, cannot use absolute paths
- Storages at a site can be listed:
  - deshl list hpcx –s
- “home” storage for user data and jobs
- “root” storage for system utilities
- Core users have access to DEISA GPFS via deisa_home and deisa_data storages

Data Staging Operations

- Upload a file from local workstation to a DEISA site
  - **`deshl copy c:/deshl/test.txt hpcx/home/test.txt`**
- Download a file to local workstation from a DEISA site
  - **`deshl copy hpcx/home/test.txt c:/deshl/test.txt`**
- List a directory or file on a DEISA site
  - **`deshl list hpcx/home/jobs`**
- List available storages on a DEISA site
  - **`deshl list hpcx -s`**
- Copy a file or directory between DEISA sites
  - **`deshl copy hpcx/home/test.txt fzj/home/test.txt`**
- Rename a file or directory on a DEISA site
  - **`deshl move hpcx/home/test.txt hpcx/home/test2.txt`**
- Move a file or directory between DEISA sites
  - **`deshl move hpcx/home/test.txt fzj/home/test2.txt`**
- Delete a file or directory on a DEISA site
  - **`deshl remove fzj/home/test2/txt`**

Job Definition

- Jobs defined by SAGA directive files
- Directives used to define job properties
- DESHL Client builds job description from directives and submits to UNICORE infrastructure as Abstract Job Object (AJO)



Simple job management scenario

Distributed European Infrastructure for Supercomputing Applications

- Executable application resides on a DEISA supercomputer
- User constructs a SAGA directives file to run the application
- User submits the job
  - deshl submit <script file>
  - returns a unique identifier for the submitted job
- User tracks the status of the job
  - deshl status <job identifier>
- When job has completed, user retrieves any stdout and stderr produced by the job
  - deshl fetch <job identifier>

Simple SAGA Directives File

```
#!/bin/bash
# Test job file for DESHL
#
# SAGA JobDefinition based directives:
#$ SAGA_JobCmd = hello.sh
#$ SAGA_FileTransfer = file:///jobs/hello.sh#HOME > hello.sh
#$ SAGA_HostList = ssl://admin.hpcx.ac.uk:4433/EPCC%20HPCx
```

Directives file on client, executable on server.

UNICORE jobs run in a temporary user space (USPACE) created for the job. Required files are staged in or results staged out using SAGA_FileTransfer directive.

In above example, actual job to be run is a script located at ssl://admin.hpcx.ac.uk:4433/EPCC%20HPCx/home/jobs/hello.sh

Supported SAGA Directives (1)

Distributed European Infrastructure for Supercomputing Applications

- SAGA_JobCmd – The job executable path
- SAGA_FileTransfer – a file transfer local to the host where the job is to run, to stage in files required by the job to its uspace or stage out data produced by the job from the uspace.
  - Note: SAGA_FileTransfer and SAGA_JobCmd are the only directives which MUST be specified to run a job
- SAGA_JobArgs – Arguments to be passed to the job to be treated as command line arguments
- SAGA_JobEnv – Set an enviroment variable for the job to use and send information to UNICORE
- SAGA_JobName – a descriptive name for the job

Supported Directives (2)

- SAGA_HostList – The destination host for the job (can also be specified at the command line)
- SAGA_NumTasks – The number of tasks
- SAGA_NumCpus – The number of threads per task
- SAGA_WallClockSoftLimit – The time the job should run for in seconds
- SAGA_Memory – policy decided by Target System Interface (TSI)

Job Management (1)

- Submit a job
  - **`deshl submit [options] <directives_script> <job arguments>`**
  - **`asynchronous`**
- Returns a unique job identifier
  - x1951302172ssl%3A%2F%2Fadmin.hpcx.ac.uk%3A4433%2FEPCC%2520HPCx
  - Very long, but has to contain encoded site information to allow persistence
  - Returned job identifier subsequently used with deshl status, terminate, fetch commands
- Supported command line options (can also be specified in SAGA directives file):
  - “n” A descriptive name for the job
  - “q” The DEISA site to which a job is to be submitted
  - “v” Enviroment variables to be passed to the job
- Example:
  - **`deshl submit –q hpcx –n “My test job” –v inputfile=“data.txt” –v outputfile=“results.txt” jobdirectives.sh`**

Retrieve job status

- Uses job identifier returned by submit command
  - **deshl status <job identifier>**
  - **Returned state is one of Running, DoneOk, Failed**
- Can return simple job status or extended job details
  - **deshl status –f <job identifier>**

Example status output

```
Example simple details:
Job:
x53329517ssl%3A%2F%2Fadmin.hpcx.ac.uk%3A4433%2FEPCC%2520HPCx
,
has status: DoneOk

Example extended details:
Site: ssl://admin.hpcx.ac.uk:4433/EPCC%20HPCx
Name: hello.sh
Running 06/19 11:14:18
DoneOk 06/19 11:24:33
```

Tracking submitted jobs

- List all current jobs
  - Looks for submitted jobs on ALL configured sites
  - Lists job identifiers
  - **`deshl jobs`**
- List jobs at a specific site
  - Only looks for jobs on the specified site
  - **`deshl jobs -q hpcx`**
- List extended job detail
  - Lists job name, status, submission time etc.
  - **`deshl jobs -f`**
  - **`deshl jobs -f -q hpcx`**

Example job listing

Distributed European Infrastructure for Supercomputing Applications

deshl list –f –q hpcx

x1327023520ssl%3A%2F%2Fadmin.hpcx.ac.uk%3A4433%2FEPCC%2520HPCx
Site: ssl://admin.hpcx.ac.uk:4433/EPCC%20HPCx
Name: hello.sh
Running 06/19 12:07:50
DoneOk 06/19 12:10:03

x53329517ssl%3A%2F%2Fadmin.hpcx.ac.uk%3A4433%2FEPCC%2520HPCx
Site: ssl://admin.hpcx.ac.uk:4433/EPCC%20HPCx
Name: hello.sh
Running 06/19 11:14:18
DoneOk 06/19 11:24:33

Terminate a job

- Terminate a running job
- Stops job from executing but does not free resources
- Any stderr or stdout produced prior to terminate can be subsequently retrieved with “fetch” command
  - **`deshl terminate <job identifier>`**

Fetch a job

- Frees any resources for a completed or terminated job and retrieves stdout and stderr produced by the job
- Completed, terminated or failed job remains on host until explicitly fetched by the user
  - **`deshl fetch <job identifier>`**
- Can retrieve output files to a specific directory
  - **`deshl fetch –d <directory> <job identifier>`**

Simple scenario (1)

Distributed European Infrastructure for Supercomputing Applications

- Server-side script, ssl://admin.hpcx.ac.uk:4433/EPCC%20HPCx/home/jobs/cat.sh takes two filenames, concatenates them to a third specified filename

```
#!/bin/bash

echo $inputfilea
echo $inputfileb
echo $outputfile

cat $inputfilea $inputfileb > $outputfile
```

cat.sh

Simple scenario (2)

Distributed European Infrastructure for Supercomputing Applications

- Client-side script to submit job
- Input and output files specified as environment variables
- Assume input data files in ssl://admin.hpcx.ac.uk:4433/EPCC%20HPCx/home/data
- File staging used to retrieve output file from USPACE
- Output file retrieved to ssl://admin.hpcx.ac.uk:4433/EPCC%20HPCx/home/data

```
#!/bin/sh
# Test job script for DESHL using SAGA.
#
# SAGA JobDefinition based directives:
#$ SAGA_FileTransfer = file:///jobs/cat.sh#HOME > cat.sh
#$ SAGA_JobCmd = cat.sh
#$ SAGA_FileTransfer = file:///data/file1.dat#HOME > file1.dat
#$ SAGA_FileTransfer = file:///data/file2.dat#HOME > file2.dat
#$ SAGA_HostList = ssl://admin.hpcx.ac.uk:4433/EPCC%20HPCx
#$ SAGA_JobEnv = inputfilea=file1.dat
#$ SAGA_JobEnv = inputfileb=file2.dat
#$ SAGA_JobEnv = outputfile=file3.dat
#$ SAGA_FileTransfer = file:///data/output.dat#HOME < file3.dat
```

Image Processing Example

- Edge detection executable, running on 16 processors, 1 thread per process

```
#!/bin/sh
#$ SAGA_FileTransfer = file:///data/edge192x360.dat#HOME > edge192x360.dat
#$ SAGA_FileTransfer = file:///jobs/imagempi.aix#HOME > imagempi.aix
#$ SAGA_FileTransfer = file:///bin/poe#ROOT > poe
#$ SAGA_HostList = ssl://admin.hpcx.ac.uk:4433/EPCC%20HPCx
#$ SAGA_NumTasks = 16
#$ SAGA_NumCpus = 1
#$ SAGA_JobCmd = poe imagempi.aix
#$ SAGA_JobName = image processing example
#$ SAGA_FileTransfer = file:///results/image.pgm#HOME < image192x360.pgm
```

DESHL Summary

Distributed European Infrastructure for Supercomputing Applications

- The DESHL software has been developed by the seventh Joint Research Activity (JRA7) of the EU-funded DEISA project. EPCC and ECMWF from the UK, FZJ from Germany and CINECA from Italy are the participants in this research activity.
- DESHL allows user and applications to manage batch jobs and data in a uniform manner regardless of underlying differences in hardware and software
- DESHL employs emerging grid standards so that users and applications are not affected by changes in underlying software
- For further information and to download the DESHL Client software:
  - http://deisa-jra7.forge.nesc.ac.uk/
- DESHL email address:
  - deisa-jra7-deshl@forge.nesc.ac.uk