

Accessing External Data Resources from DEISA Grid

Nicola Mc Donnell, EPCC
nix@epcc.ed.ac.uk



Accessing External Data Resources from DEISA GRID

Overview

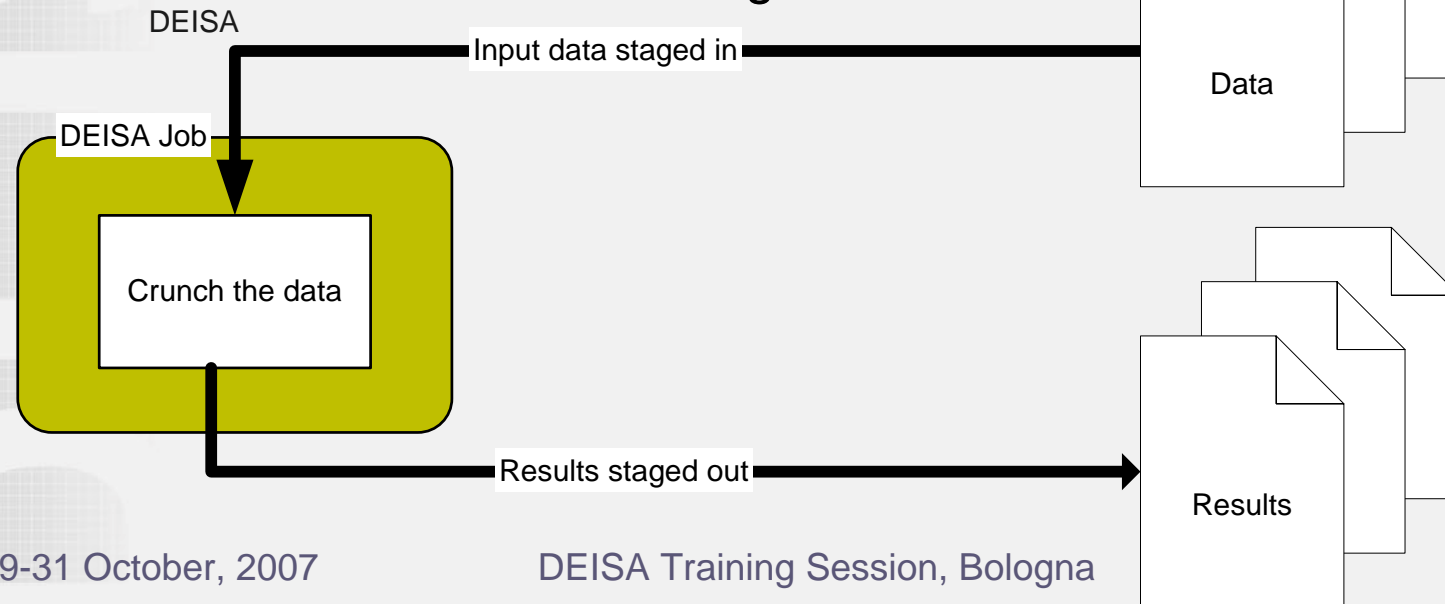
- What do we mean?
- Prerequisite
- The gory details
- What next

What does it all mean?

Classic HPC approach



- Data is in **flat files**
- **DEISA GPFS** hosts the standard databases.
- If not :
 - Stage in input data
 - Stage out results

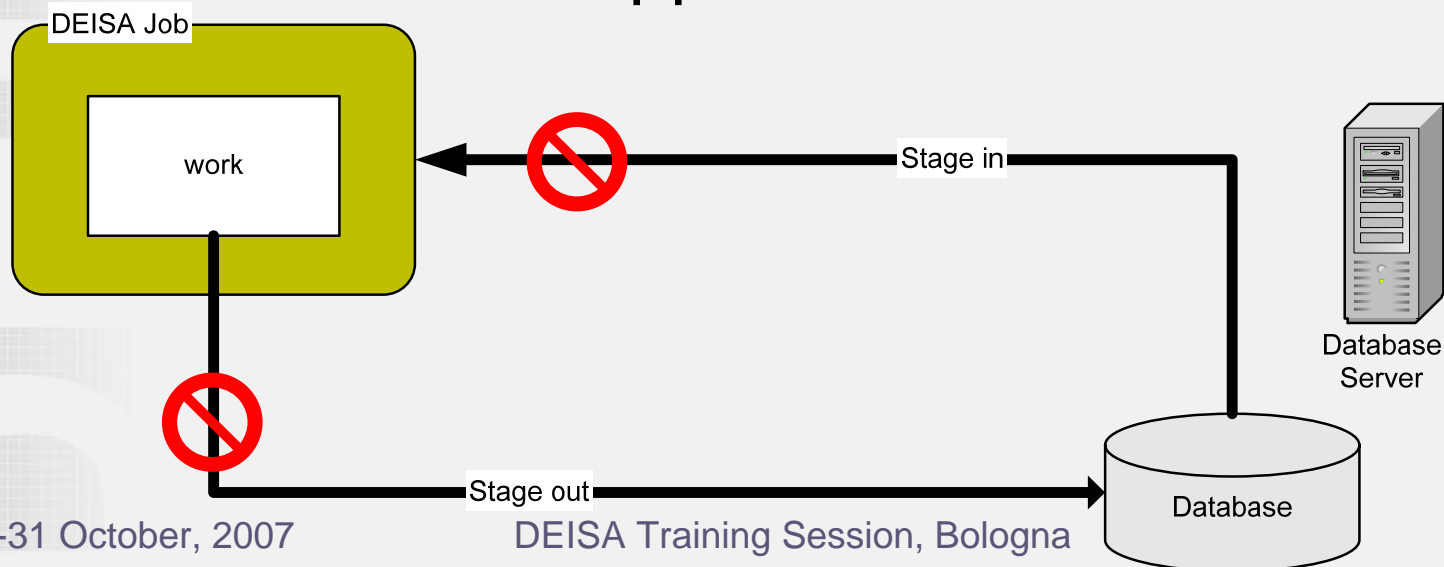


What about data in Relational or XML database

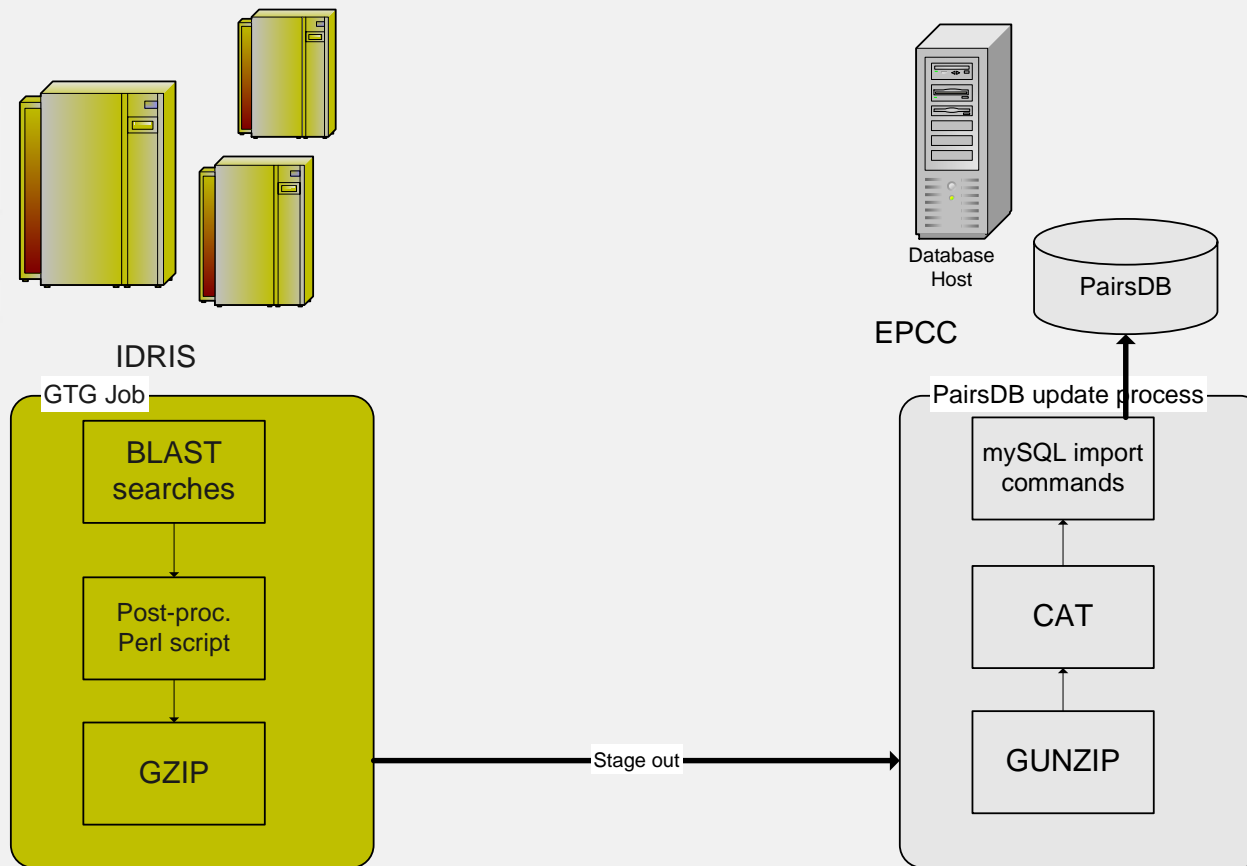


DEISA

Can't stage data in a relational or XML database application in or out

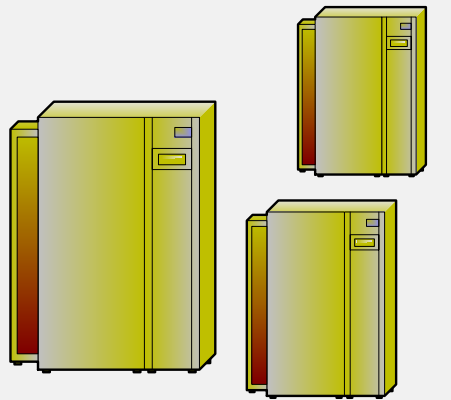


Case Study – DECI Project GTG

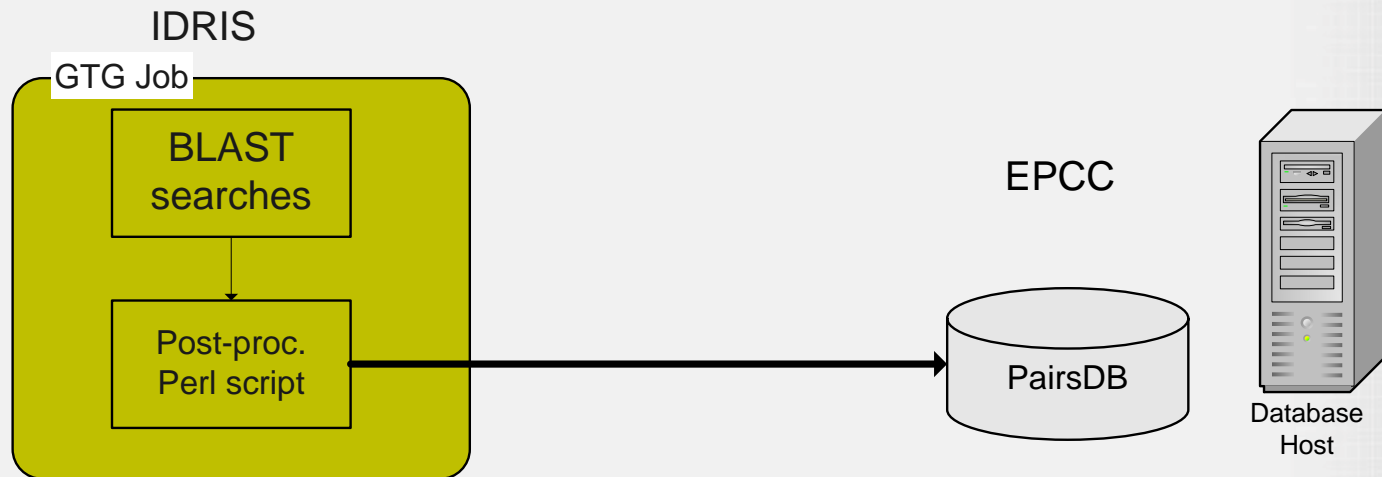


22,000 GTG jobs running at IDRIS, each job running a batch of 50 searches. The results of these searches that are stored in the PairsDB database at EPCC.

Case Study – DECI Project GTG



If only we could access
the database directly?



Use a database access library

For example, Perl DBD:mysql module

```
use DBI;  
$dbh = DBI->connect($dsn, $user, $password);  
$sth = $dbh->prepare("INSERT nrdb40 VALUES  
    (371, 2982826, -232.5211, 1, 712, +396-2+109-3+207, 1)");
```

Use a database access library

X Installation

Non-trivial

X Firewall Issue

Firewall at EPCC needs to allow connections from each of the IDRIS execute nodes to the Database server.

X Grant Issue

Configure the grant permissions in PairsDB database to allow connections from each of the IDRIS execute nodes.

Use a database client tool

For example, mysql client tool

```
mysql --execute="INSERT nrdb40 VALUES  
(371, 2982826, -232.5211, 1, 712, +396-2+109-3+207, 1) "
```

Use a database client tool

✓ Installation

Easier to install

✗ Firewall Issue

Firewall at EPCC needs to allow connections from each of the IDRIS execute nodes to the Database server.

✗ Grant Issue

Configure the grant permissions in PairsDB database to allow connections from each of the IDRIS execute nodes.

Use Java JDBC

For example, MySQL Connector/J

```
DriverManager.getConnection( "jdbc:odbc:Database" ) ;  
Statement stmt = conn.createStatement() ;  
ResultSet rs = stmt.executeQuery( " INSERT nrdb40 VALUES  
    (371, 2982826, -232.5211, 1, 712, +396-2+109-3+207, 1) " ) ;
```

Use Java JDBC

Installation

Java based, really easy to install

Firewall Issue

Firewall at EPCC needs to allow connections from each of the IDRIS execute nodes to the Database server.

Grant Issue

Configure the grant permissions in PairsDB database to allow connections from each of the IDRIS execute nodes.

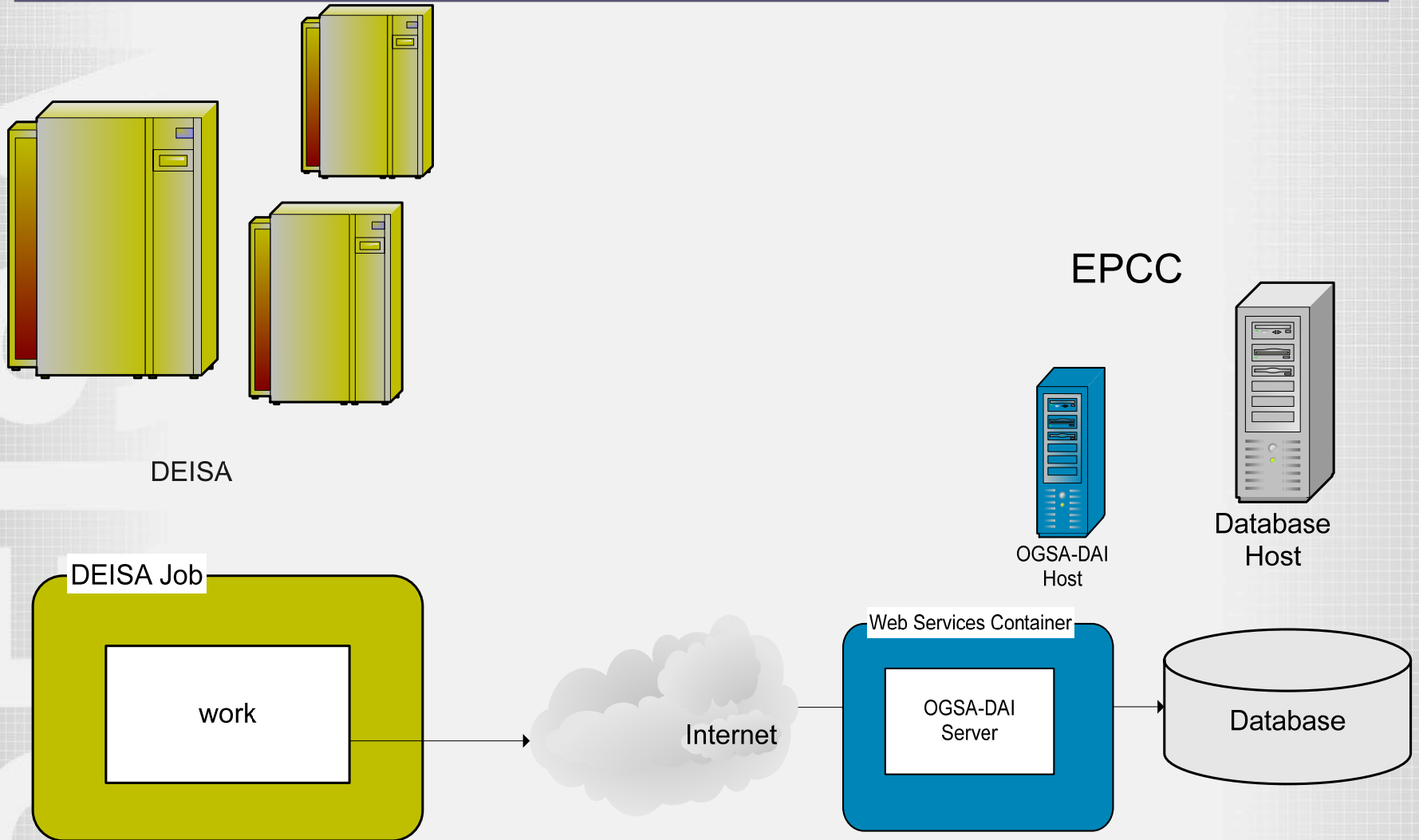
What is OGSA-DAI

OGSA-DAI (Open Grid Services Architecture - Data Access and Integration) is a



- client toolkit and
- extensible server
- for building data-centric workflows
- for accessing, integrating, transforming and delivering data
- within a Grid

How would it fit in?



Use OGSA-DAI

```
java uk.org.ogsadai.client.toolkit.example.SQLClient \  
-u http://ogsadai.epcc.ed.ac.uk:8080/dai/services/ \  
-e DataRequestExecutionResource \  
-d PairsDB \  
-q "INSERT nrdb40 VALUES  
      (371, 2982826, -232.5211, 1, 712, +396-2+109-3+207, 1)"
```

Use a database client tool

✓ Installation

Java based; really easy to install.

- have to install a server near the database

✓ Firewall Issue

The Firewall is opened to the OGSA-DAI server which usually listens on the standard port 8080

✓ Grant Issue

The grant permissions in PairsDB database to allow connections from ogsadai.epcc.ed.ac.uk.

Prerequisites

DEISA user

Submit jobs using either:

– **DESHL**

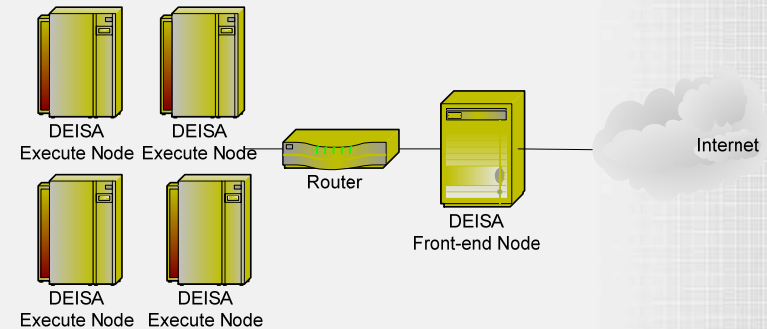
- Command line job submission tool

– **UNICORE**

- Graphical job submission tool

DEISA Execution site

Route to the internet from
the DEISA execute node.

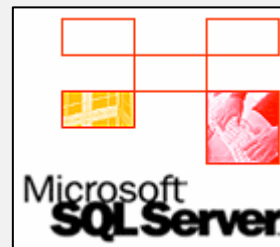


Java accessible to the
execute nodes

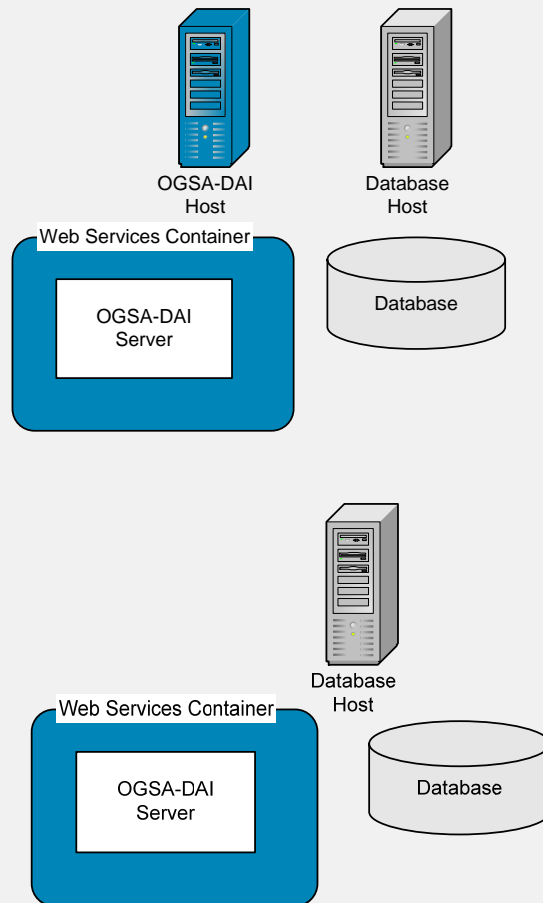


Relational of XML Database

Relational or XML database
supported by OGSA-DAI



OGSA-DAI Host



Host 'close to' the database host where the OGSA-DAI server can be installed.

The OGSA-DAI host can be the database host

The Gory Details

The good news

- You only have to do this once...
- One OGSA-DAI server can server many different databases
- Details are in the handout
- Need help contact DEISA support at your home site for assistance

Check the Java version

Shell script

Write a shell script to get the Java version

```
#!/bin/bash  
  
module load deisa java  
  
# Check the version of java which is found.  
java -version
```

DESHL Job script

Write a DESHL job submission script to run the shell script.

```
#!/bin/bash

# SAGA JobDefinition based directives:
#$ SAGA_JobName = JavaTest
#$ SAGA_FileTransfer = file:///java_test.sh#DEISA_HOME > java_test.sh
#$ SAGA_JobCmd = java_test.sh
#$ SAGA_NumCpus = 1
#$ SAGA_NumTasks = 8
#$ SAGA_WallClockSoftLimit=60
```

Submit the job

Copy the shell script to IDRIS:

```
> deshl copy -f java_test.sh IDRIS/deisa_home/java_test.sh
```

Use DESHL to submit the job:

```
> deshl submit -q IDRIS java_test_submit.sh  
Your job: <job-id>, has been successfully submitted.
```

Retrieve the results

```
> deshl status <job-id>
```

```
Job: <job-id>, has status: Done, Unicore:FINISHED
```

```
> deshl fetch <job-id>
```

```
> cat <job-id>.err
```

```
This script was created and executed by Unicore
```

```
UNICORE - start of user output on stderr
```

```
java version "1.4.2"
```

```
Java(TM) 2 Runtime Environment, Standard Edition (build 1.4.2)
```

```
Classic VM (build 1.4.2, J2RE 1.4.2 IBM AIX build ca142-20060421 (SR5) (JIT  
enabled: jitc))
```

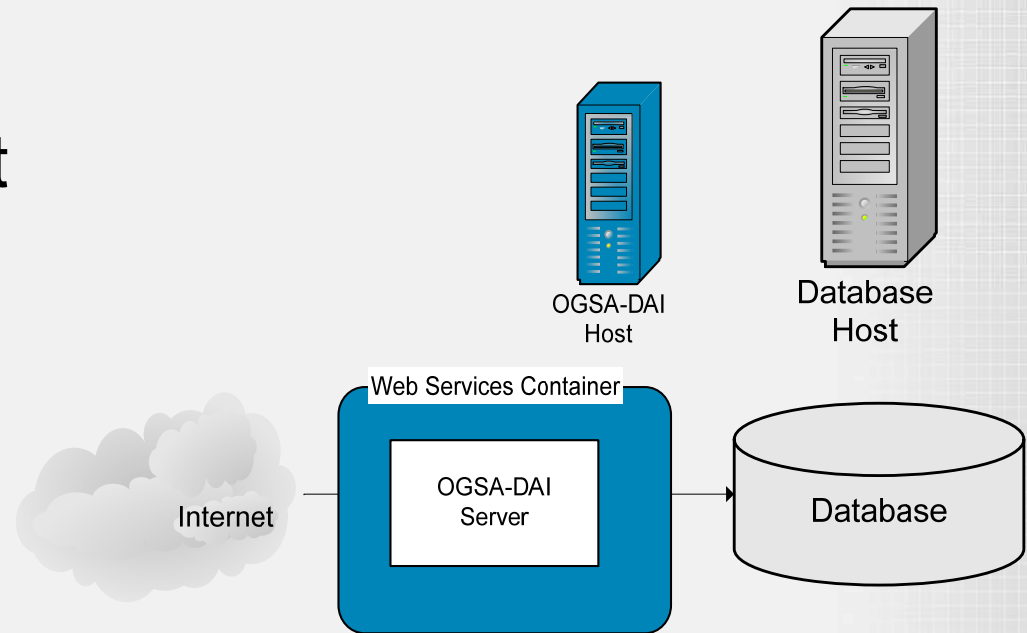
```
UNICORE - end of user output on stderr
```

```
UNICORE EXIT STATUS 0 +
```

Set up the OGSA-DAI Server

Install the OGSA-DAI server

- **Download** OGSA-DAI from the OGSA-DAI project website.
- **Install** it on your chosen.
- **Register** your database as a Data Resource.



Check the network connection

ConnectionTest.java

Write a Java program which reads the OGSA-DAI Service Description from the server:

```
URL url =
    new URL("http:// ogsadai.epcc.ed.ac.uk:8080/dai/services/Version?wsdl" );
InputStreamReader input = new InputStreamReader(url.openStream());
...
while ((readBytes = input.read(charArray)) >= 0) {
    buf.append(charArray, 0, readBytes);
}
System.out.println( "Successfully retrieved the OGSA-DAI service
    Description." );
```

Follow the usual steps

Write a shell script to execute the
ConnectionTest.class

Write a DESHL Script to submit the job

Submit our job and when completed we
retrieve the results

Success

View the jobs std out file to see if we were successful:

```
> cat <job-id>.out
```

This script was created and executed by Unicore

UNICORE - start of user output on stdout

Successfully retrieved the OGSA-DAI service Description.

UNICORE - end of user output on stdout

Failure

If there is an error double check the OGSA-DAI host and port number.

```
> cat <job-id>.err
```

```
This script was created and executed by Unicore
```

```
UNICORE - start of user output on stderr
```

```
Failed to retrieve the Version WSDL for the OGSA-DAI service.
```

```
Check that the OGSA-DAI Service URL,
```

```
    http://ogsadai.epcc.ed.ac.uk:8080/dai/services/Version?wsdl, is a valid URL.
```

```
UNICORE - end of user output on stderr
```

```
UNICORE EXIT STATUS 1 +
```

Install the OGSA-DAI Client

Install the OGSA-DAI client

Copy up the client zip archive:

```
> desh1 copy -f ogsadai-3.0-axis-1.4-bin-extended-jre.zip  
IDRIS/deisa_home/ogsadai-3.0-axis-1.4-bin-extended-jre.zip
```

Write a script to unzip the archive:

```
#!/bin/bash  
Module load deisa  
  
# Move to the home directory.  
cd $DEISA_HOME  
  
# Unzip the runtime.  
unzip ogsadai-3.0-axis-1.4-bin-extended-jre.zip
```

Retrieve and view the results

Viewing the results

```
> cat <job-id>.out
```

This script was created and executed by Unicore

UNICORE - start of user output on stdout

Archive: ogsadai-3.0-axis-1.4-bin-extended-jre.zip

creating: ogsadai-3.0-axis-1.4-bin-extended-jre/

inflating: ogsadai-3.0-axis-1.4-bin-extended-jre/lib/addressing-1.0.jar

and so on ...

UNICORE - end of user output on stdout

Recap

- Checked Java in the DEISA execute site.
- Installed the OGSA-DAI server near the database.
- Checked the network connection to the OGSA-DAI server from the DEISA execution site.
- Installed the OGSA-DAI client on the DEISA execution site.

Run the OGSA-DAI Client

SQLClient Java program

Run the SQLClient:

```
#!/bin/bash
export OGSADAI_RUNTIME_HOME=ogsadai-3.0-axis-1.4-bin-extended-jre
export OGSADAI_CLASSPATH=\
$OGSADAI_RUNTIME_HOME/\
$OGSADAI_RUNTIME_HOME/lib/activation.jar:\
And so on ...
```

```
java -cp $OGSADAI_CLASSPATH:. \
uk.org.ogsadai.client.toolkit.example.SQLClient \
-u http://ogsadai.epcc.ed.ac.uk:8080/dai/services/ \
-e DataRequestExecutionResource \
-d PairsDB \
-q "SELECT * FROM nrdb40"
```

View the results

```
> cat <job-id>.out
```

This script was created and executed by Unicore

UNICORE - start of user output on stdout

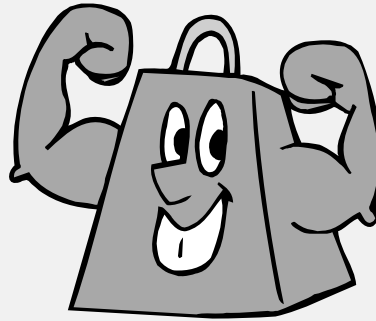
one	two	three	four	five	six	seven
371	2982826	-232.5211	1	712	+396-2+109-3+207	1
371	2982827	-236.5611	1	712	+396-2+108-3+207	1
371	2982828	-252.5611	1	712	+396-2+209-3+207	1
371	2982829	-232.5611	1	712	+395-2+109-3+207	1

and so on

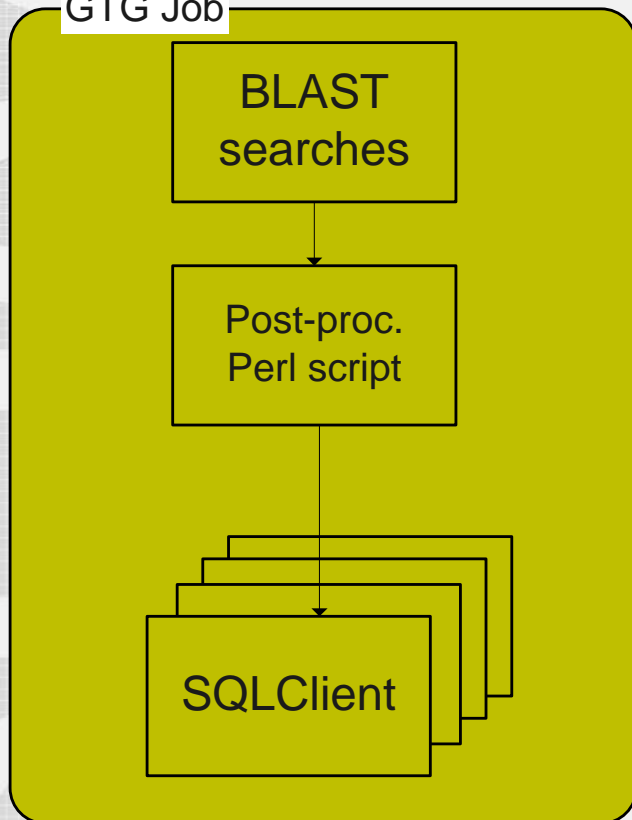
UNICORE - end of user output on stdout

What next?

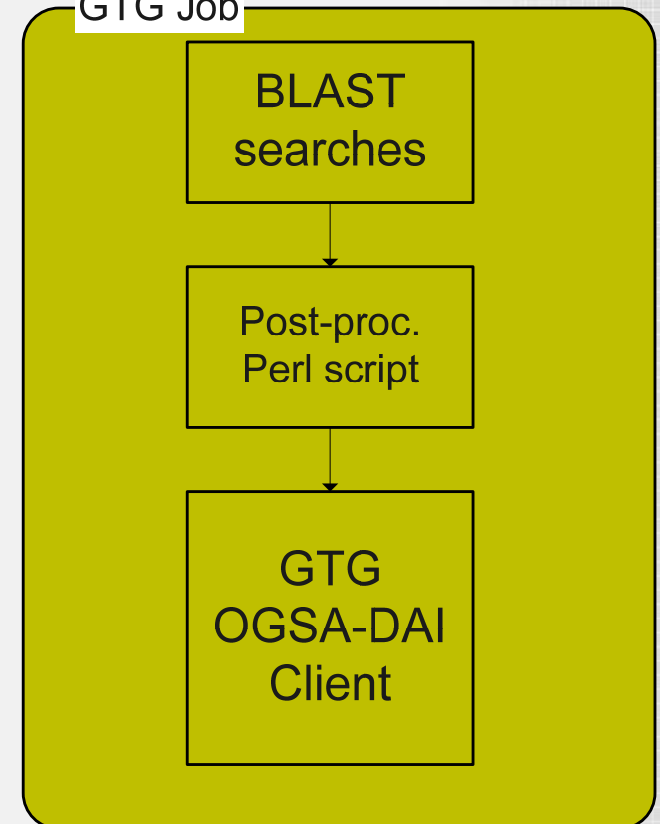
Custom OGSA-DAI client



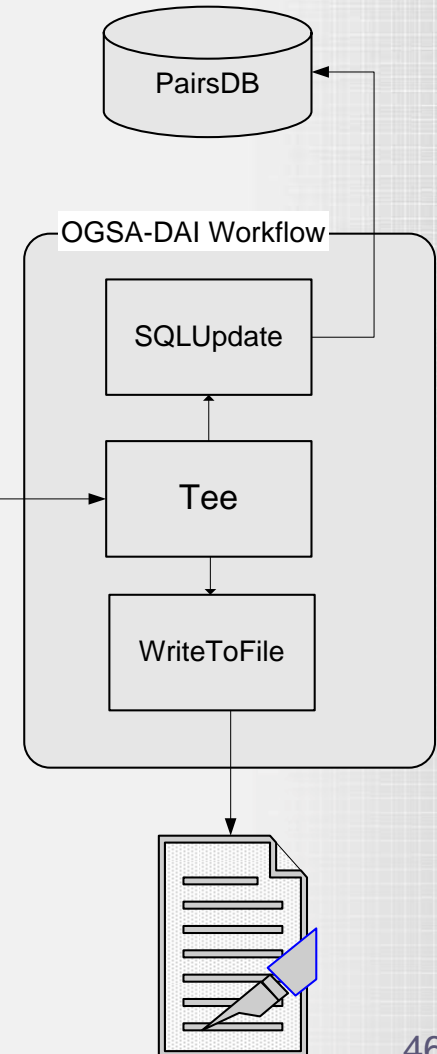
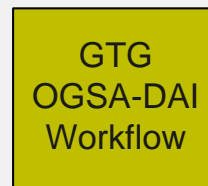
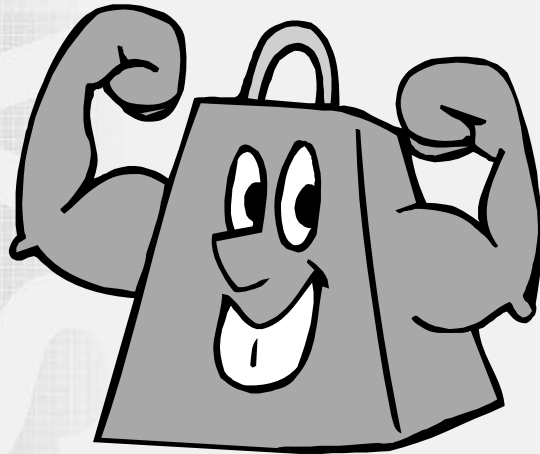
GTG Job



GTG Job

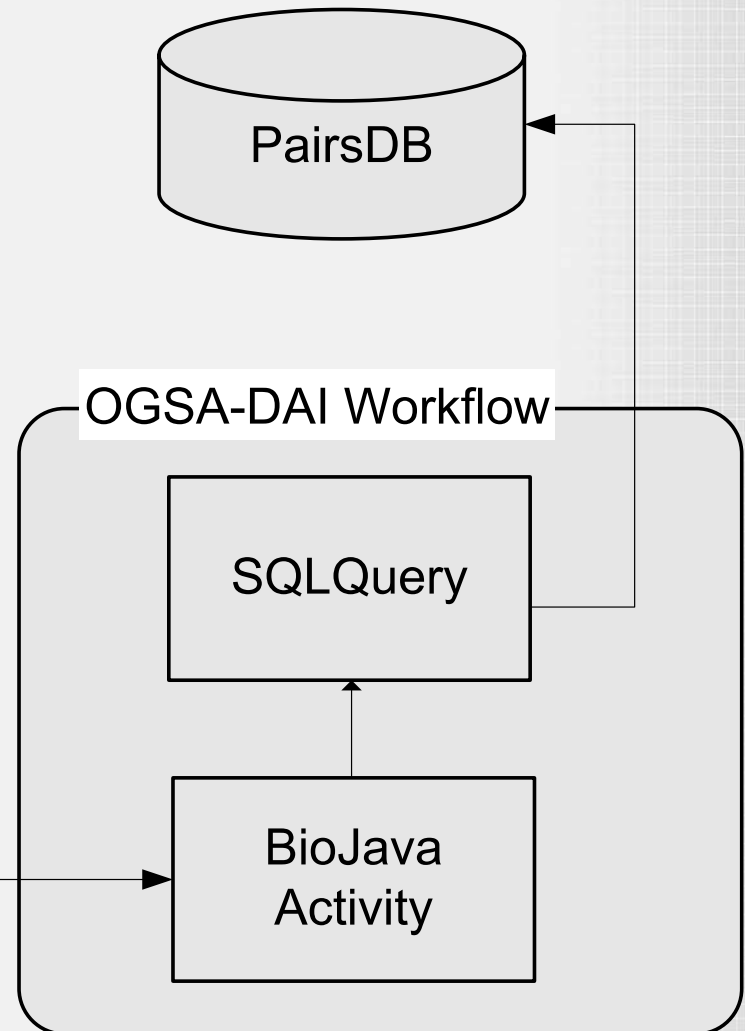
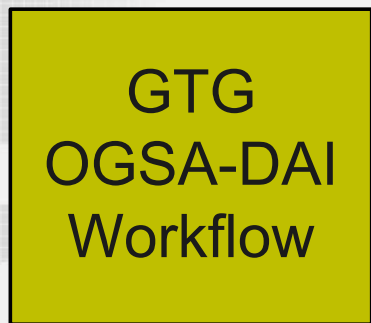
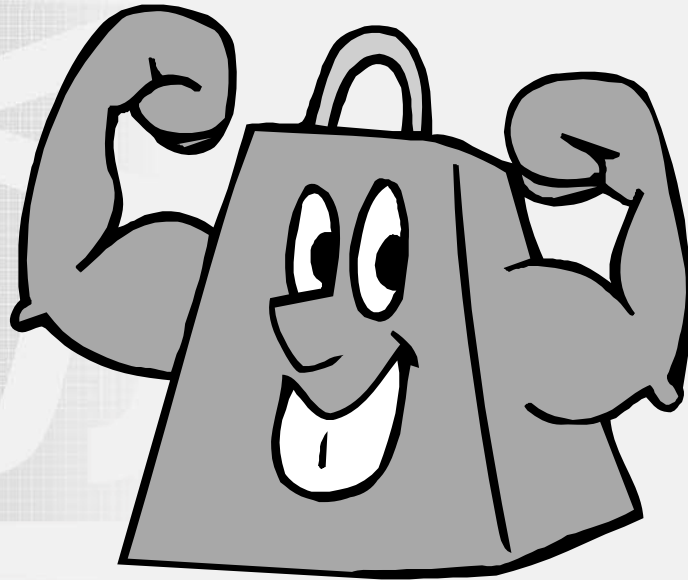


A complex workflow



- The data is duplicated using the Tee activity on the server-side
- One set is sent to database for update
- Second set is parsed and logged to a file

Develop an OGSA-DAI plug-in



Handout and src for this session

- <http://www.deisa.org/training/Bologna2007>

Visit the OGSA-DAI website

- <http://www.ogsadai.org.uk/>

OGSA-DAI Training sessions

- Deploying Grid Data Services using OGSA-DAI, 1-2 November, NeSC Edinburgh

DEISA User Support

- <http://www.deisa.org/userscorner/>



Accessing External Data Resources from DEISA GRID

Summary

- Accessing data in a relational or XML database from within a DEISA job.
- Pros and Cons of different approaches
- Prerequisite to using OGSA-DAI
- Setting up the OGSA-DAI server and client
- The next steps.....

Thank you

`nix@epcc.ed.ac.uk`

The source code for this presentation can be downloaded from
<http://www.deisa.org/training/Bologna2007>