NGS

National Grid Service

# GridSAM: an Introduction

Mike Mineter

# Job submission with JSDL and GridSAM

- To guide us through the acronyms… selection of slides given by A. Stephen McGough (Imperial College London) in Edinburgh on 22 Feb 2007 http://indico.cern.ch/conferenceDisplay.py?confId=12549
- Extra information is in hidden slides

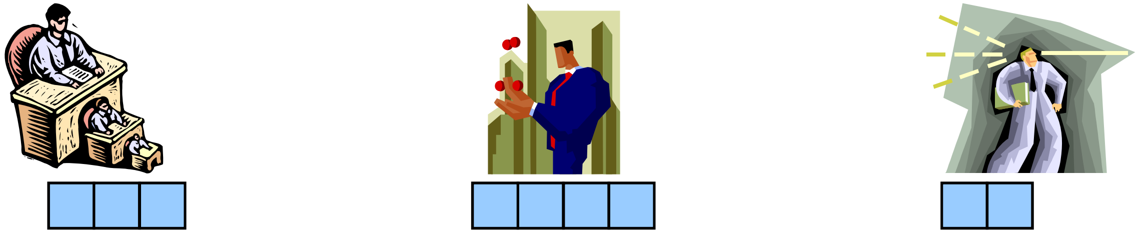
# Outline

- Overview
- Other Way
  - JSDL
  - GridSAM

# Overview

# Running Jobs on the Grid

# Context

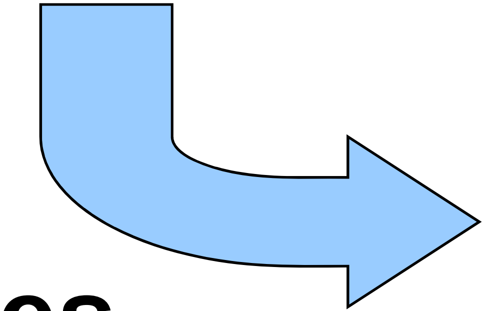


**jobs / legacy code / binary executables**

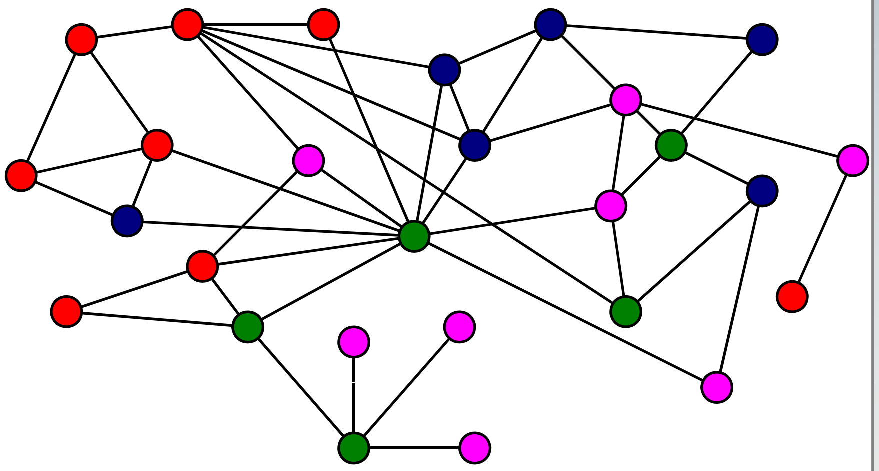
**Middleware**

**Map to**

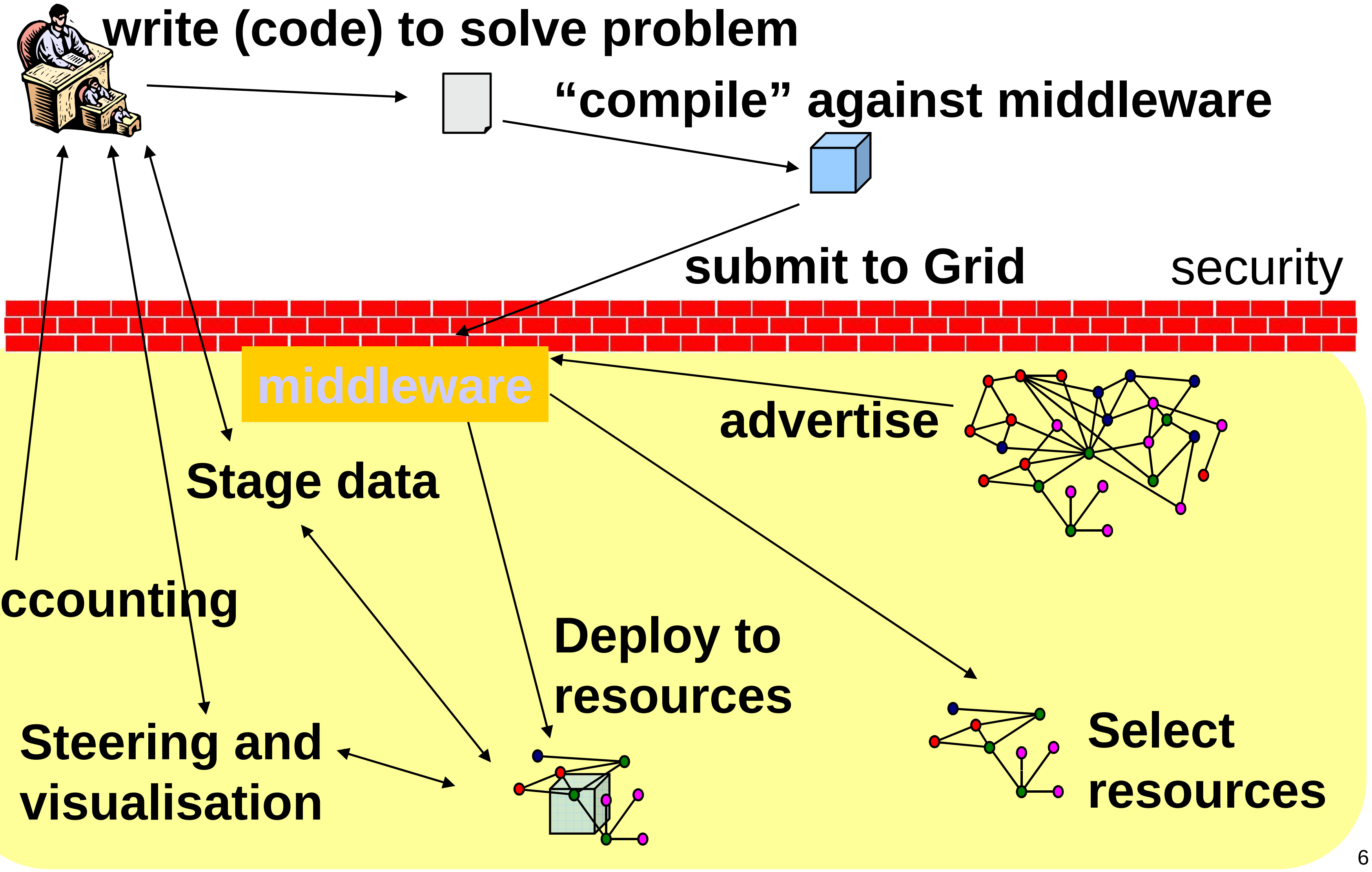
**resources**



**Resources**



# Stages to using the Grid – Classical View



write (code) to solve problem

“compile” against middleware

submit to Grid

security

middleware

advertise

Stage data

accounting

Deploy to resources

Select resources

Steering and visualisation

# What is wrong with this picture?

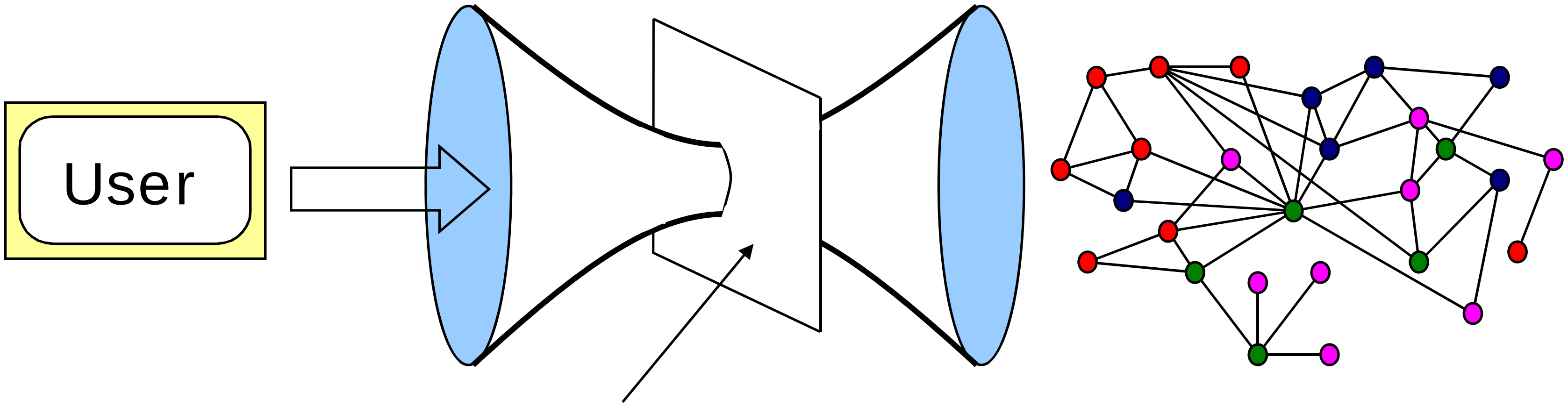
There are already many DRM systems (Condor, Globus…)

Why do we need another one?

- We don't. What we really need is for them all to be able to talk to each other
  - Make life easy for all
- We need a service which makes systems look the same

# To make life easy

We want to hide the heterogeneity of the Grid



Hide heterogeneity by tight abstraction here

Grid resources

# Other Way…

# Standards Based Job Submission

# If all DRM systems supported the same interface…

If we had:

- One interface definition for job submission
- One job description language

Then life would be easier!

We're getting there

- JSDL is a proposed standard job submission description language
- OGSA-BES are proposing a basic execution service interface

One day hopefully everyone will support this

- Till then…

London e-Science Centre

# JSDL 1.0 Primer

Ali Anjomshoaa, Fred Brisard, Michel Drescher,
Donal K. Fellows, William Lee, An Ly, Steve McGough,
Darren Pulsipher, Andreas Savva, Chris Smith

# JSDL Introduction

JSDL stands for *Job Submission Description Language*

- A language for *describing the requirements of computational jobs for submission* to Grids and other systems.

A JSDL *document* describes the job requirements

- *What to do, not how to do it*
- No Defaults
- All elements must be satisfied for the document to be satisfied

JSDL *does not* define a submission interface or what the results of a submission look like

JSDL 1.0 is published as GFD-R-P.56

- Includes description of JSDL elements and XML Schema
- Available at http://www.ggf.org/gf/docs/?final

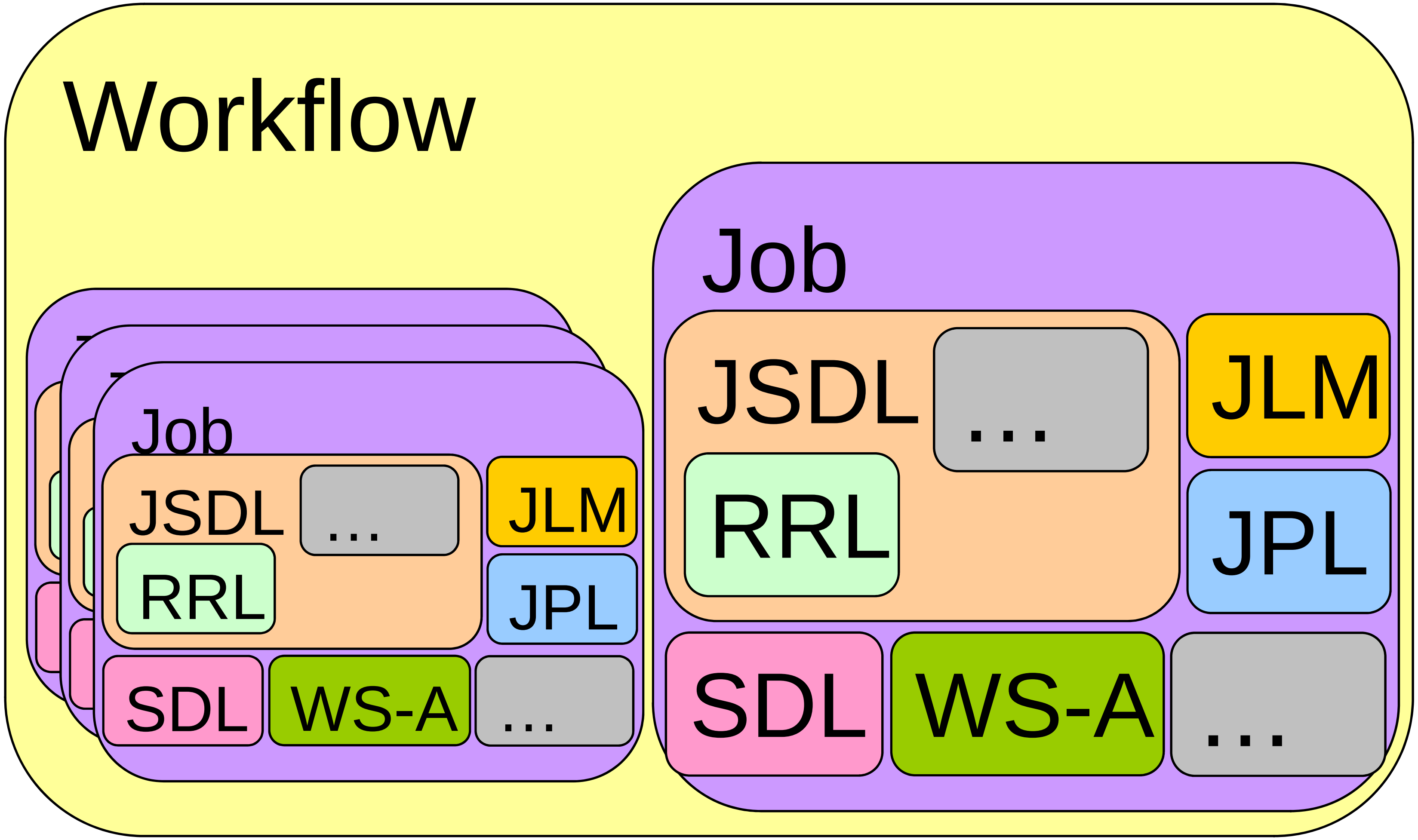
# JSDL Document

- A JSDL document is an XML document
- It may contain
  - Generic (job) identification information
  - Application description
  - Resource requirements (main focus is computational jobs)
  - Description of required data files
- It is a template language
- Open content language – compose-able with others
- Out of scope, for JSDL version 1.0:
  - Scheduling, Workflow,Security …

London e-Science Centre

Imperial College London

# JSDL: Conceptual relation with other standards



RRL – Resource Requirements Language
SDL – Scheduling Description Language
WS-A – WS-Agreement
JLM – Job Lifetime Management
JPL – Job Policy Language

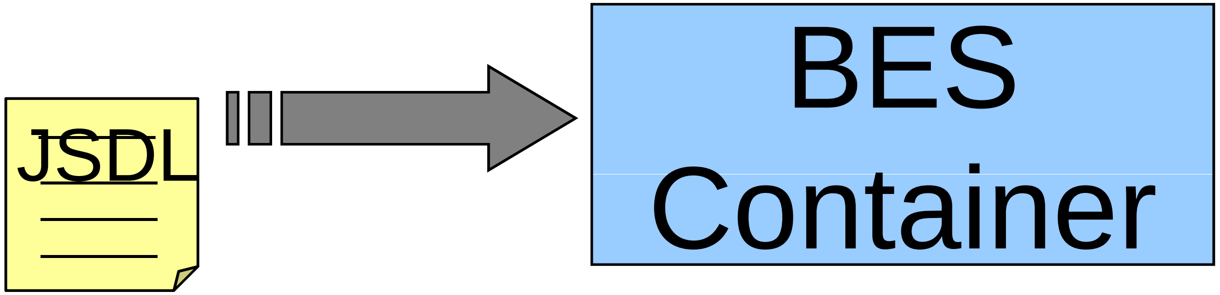
# A few words on JSDL and BES

JSDL is a language

- No submission interface defined (on purpose)
- JSDL is independent of submission interfaces

BES is defining a Web Service interface which consumes JSDL documents

- This is not the only use of JSDL
- Though we do like it 😎



# Hidden slides give more detail!

# JSDL Document Structure Overview

```
<JobDefinition>
   <JobDescription>
      <JobIdentification ... />?
      <Application ... />?
      <Resources... />?
      <DataStaging ... />*
   </JobDescription>
</JobDefinition>
```

Note:

| | |
|---|---|
| None | [1..1] |
| ? | [0..1] |
| * | [0..n] |
| + | [1..n] |

# Job Identification Element

```
<JobIdentification>
   <JobName ... />?
   <Description ... />?
   <JobAnnotation ... />*
   <JobProject ... />*
   <xsd:any##other>*
</JobIdentification>?
```

**Example:**

```
<jsdl:JobIdentification>
   <jsdl:JobName>
      My Gnuplot invocation
   </jsdl:JobName>
   <jsdl:Description>
      Simple application ...
   </jsdl:Description>

   <tns:AAId>3452325707234
   </tns:AAId>

</jsdl:JobIdentification>
```

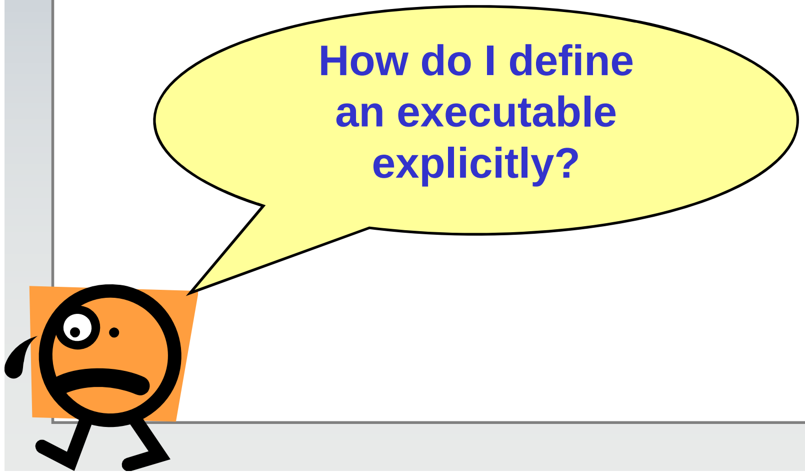
Extensibility point

# Application Element

```
<Application>
   <ApplicationName ... />?
   <ApplicationVersion ... />?
   <Description ... />?
          <xsd:any##other>*
</Application>
```

**Example:**

```
<jsdl:Application>
   <jsdl:ApplicationName>
      gnuplot
   </jsdl:ApplicationName>
   <jsdl:ApplicationVersion>
      5.7
   </jsdl:ApplicationVersion>
   <jsdl:Description>
      Use the gnuplot application v5.7
      regardless where it is installed on
      the target system
   <jsdl:Description>
</jsdl:Application>
```



# Application: POSIXApplication extension

```
<POSIXApplication>
  <Executable ... />
  <Argument ... />*
  <Input ... />?
  <Output ... />?
  <Error ... />?
  <WorkingDirectory ... />?
  <Environment ... />*
  …
</POSIXApplication>
```

POSIXApplication is a normative JSDL extension

Defines standard POSIX elements

- stdin, stdout, stderr
- Working directory
- Command line arguments
- Environment variables
- POSIX limits (not shown here)

# Hello World

```
<?xml version="1.0" encoding="UTF-8"?>
<jsdl:JobDefinition
    xmlns:jsdl=“http://schemas.ggf.org/2005/11/jsdl”
    xmlns:jsdl-posix=
              “http://schemas.ggf.org/jsdl/2005/11/jsdl-posix”>
<jsdl:JobDescription>
    <jsdl:Application>
        <jsdl-posix:POSIXApplication>
            <jsdl-posix:Executable>
                /bin/echo
            <jsdl-posix:Executable>
            <jsdl-posix:Argument>hello</jsdl-posix:Argument>
            <jsdl-posix:Argument>world</jsdl-posix:Argument>
        </jsdl-posix:POSIXApplication>
    </jsdl:Application>
  </jsdl:JobDescription>
</jsdl:JobDefinition>
```

# Resource description requirements

- Support *simple* descriptions of resource requirements
  - NOT a comprehensive resource requirements language
  - Avoided explicit heterogeneous or hierarchical descriptions
  - Can be extended with other elements for richer or more abstract descriptions
- Main target is compute jobs
  - CPU, Memory, Filesystem/Disk, Operating system requirements
- Allow some flexibility for aggregate (*Total**) requirements
  - “I want 10 CPUs in total and each resource should

# Resources Element

```
<Resources>
   <CandidateHosts ... />?
   <FileSystem .../>*
   <ExlusiveExecution .../>?
   <OperatingSystem .../>?
   <CPUArchitecture .../>?
   <IndividualCPUSpeed .../>?
   <IndividualCPUTime .../>?
   <IndividualCPUCount .../>?
   <IndividualNetworkBandwidth .../>?
   <IndividualPhysicalMemory .../>?
   <IndividualVirtualMemory .../>?
   <IndividualDiskSpace .../>?
   <TotalCPUTime .../>?
   <TotalCPUCount .../>?
   <TotalPhysicalMemory .../>?
   <TotalVirtualMemory .../>?
   <TotalDiskSpace .../>?
   <TotalResourceCount .../>?
   <xsd:any##other>*
</Resources>*
```

**Example:**
One CPU and at least 2 Megabytes of memory

```
<jsdl:Resources>
    <jsdl:CPUCount>
        <Exact> 1.0 <Exact>
    </jsdl:CPUCount>
    <jsdl:PhysicalMemory>
        <LowerBoundedRange>
            2097152.0
        </LowerBoundedRange>
    </jsdl:PhysicalMemory>
</jsdl:Resources>
```

# Relation of Individual* and Total* Resources elements

It is possible to combine Individual* and Total* elements to specify complex requirements

“I want a total of 10 CPUs, 2 or more per resource”

```
<jsdl:Resources>
    ...
    <jsdl:IndividualCPUCount>
       <jsdl:LowerBoundedRange>2.0</jsdl:LowerBoundedRange>
    </jsdl:IndividualCPUCount>
    <jsdl:TotalCPUCount>
          <jsdl:exact>10.0</jsdl:exact>
    </jsdl:TotalCPUCount>
    ...
</jsdl:Resources>
```

Caveat: Not all Individual/Total combinations make sense

# RangeValues

Define *exact* values (with an optional “*epsilon*” argument), left-open or right-open *intervals* and *ranges*.

**Example:**
Between 512MB and 2GB of memory (inclusive)

```
<jsdl:PhysicalMemory>
  <jsdl:Range>
     <jsdl:LowerBound>
              536870912.0
     </jsdl:LowerBound>
     <jsdl:UpperBound>
              2147483648.0
     </jsdl:UpperBound>
  </jsdl:Range>
</jsdl:PhysicalMemory>
```

**Example:**
Between 2 and 16 processors

```
<jsdl:IndividualCPUCount>
  <jsdl:LowerBoundedRange>
     2.0
  </jsdl:LowerBoundedRange>
  <jsdl:UpperBoundedRange>
     16.0
  </jsdl:UpperBoundedRange>
</jsdl:IndividualCPUCount>
```

# JSDL Type Definitions Example: OperatingSystemTypeEnumeration

JSDL defines a small number of types

- As far as possible re-use existing standards

Example: OperatingSystemTypeEnumeration

- Basic value set defined based on CIM:
  - Windows_XP, JavaVM, OS_390, LINUX, MACOS, Solaris, …
- CIM defines these as numbers; JSDL provides an XML definition
- Watching WS-CIM work

# Data Staging Requirement

## Previous statements included:

"A JSDL *document* describes the job requirements

*What to do, not how to do it*"

"Workflow is out of scope."

***But*** **… data staging is a common requirement for any meaningful job submission**

**Especially for batch job submission**

**No standard to describe such data movements**

## Our solution

**Assume simple model:**

**Stage-in – *Execute* – Stage-Out**

**Files required for execution**

**Files are staged-in before the job can start executing**

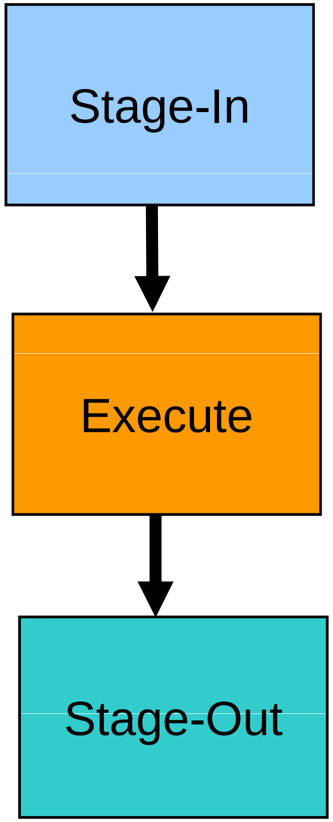
**Files to preserve**

**Files are staged-out after the job finishes execution**

## More complex approaches can be used

**But this is outside JSDL**

**You don't need to use the JSDL Data Staging**



# DataStaging Element

```
<DataStaging>
  <FileName ... />
  <FileSystemName ... />?
  <CreationFlag ... />
  <DeleteOnTermination ... />?
  <Source ... />?
  <Target ... />?
</DataStaging>*
```

**Example:**

Stage in a file (from a URL) and name it “control.txt”. In case it already exists, simply overwrite it. After the job is done, delete this file.

```
<jsdl:DataStaging>
    <jsdl:FileName>
        control.txt
    </jsdl:FileName>
    <jsdl:Source>
       <jsdl:URI>
                http://foo.bar.com/~me/control.txt
       </jsdl:URI>
    </jsdl:Source>
    <jsdl:CreationFlag>
        overwrite
    </jsdl:CreationFlag>
    <jsdl:DeleteOnTermination>
        true
    </jsdl:DeleteOnTermination>
</jsdl:DataStaging>
```

# JSDL Adoption

The following projects have presented at GGF JSDL sessions and are known to have implementations of some version of JSDL; not necessarily 1.0.

- Business Grid
- Grid Programming Environment (GPE)
- GridSAM
- HPC-Europa
- Market for Computational Services
- NAREGI
- UniGrids

The following groups also said they are or will be implementing JSDL:

- DEISA
- GridBus Project (see OGSA Roadmap, section 8)
- gridMatrix (Cadence) (presentation)
- Nordugrid

Also within GGF a number of groups either use directly or have a strong interest or connection with JSDL:

- BES-WG, CDDLM-WG, DRMAA-WG, GRAAP-WG, OGSA-WG, RSS-WG

An up-to-date version of this list is on Gridforge:

https://forge.gridforum.org/projects/jsdl-wg/document/JSDL-Adoption/en/

# JSDL Mappings

- ARC (NorduGrid)
- Condor
- eNANOS
- Fork
- Globus 2
- GRIA provider
- Grid Resource Management System (GRMS)
- JOb Scheduling Hierarchically (JOSH)
- LSF
- Sun Grid Engine
- Unicore
- <*Your mapping here*>

# *GridSAM*

Job Submission and Monitoring Web Service

Other way…

open middleware infrastructure institute uk
www.omii.ac.uk

GridSAM
London e-Science Centre

# What is GridSAM?

- A Job Submission and Monitoring Web Service
- Funded by the Open Middleware Infrastructure Institute (OMII-UK) managed programme
- Available as part of the OMII-UK release
- Open source (BSD)
- One of the first system to support the GGF Job Submission Description Language (JSDL)

# GridSAM Overview

## **Grid** Job **S**ubmission **a**nd **M**onitoring Service

What is GridSAM to the resource owners?

- A Web Service to expose heterogeneous execution resources uniformly
  - Single machine through ***Forking*** or ***SSH***
  - ***Condor Pool***
  - ***Grid Engine 6*** through ***DRMAA***
  - ***Globus 2.4.3*** exposed resources
  - OR use our plug-in API to implement …

# GridSAM Overview

**Grid** Job **S**ubmission **a**nd **M**onitoring Service

## What is GridSAM to end-users?

A set of end-user tools and client-side APIs to interact with a GridSAM web service

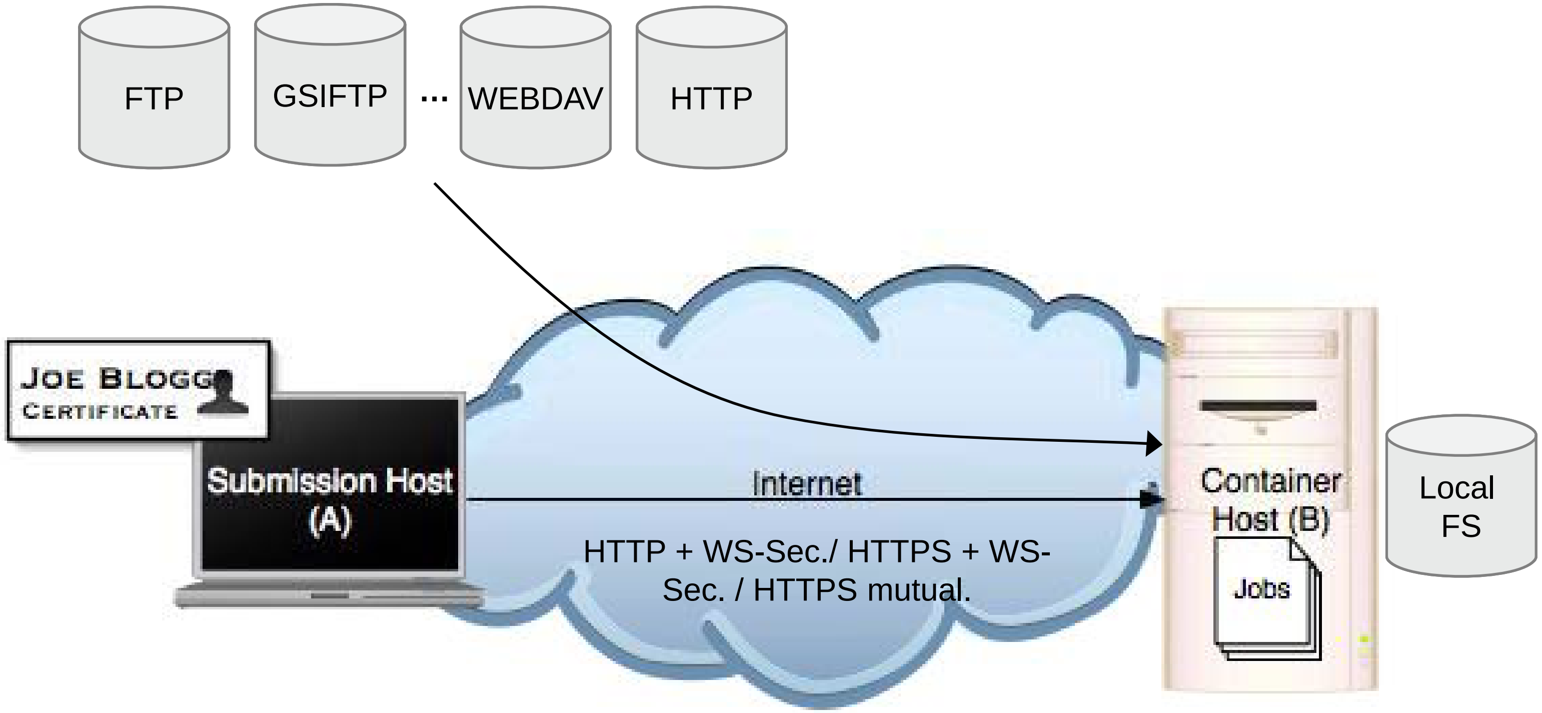
- Submit and Start Jobs
- Monitor Jobs
- Terminate Jobs
- File transfer
- Client-side submission scripting
- Client-side Java API

# What’s it not?

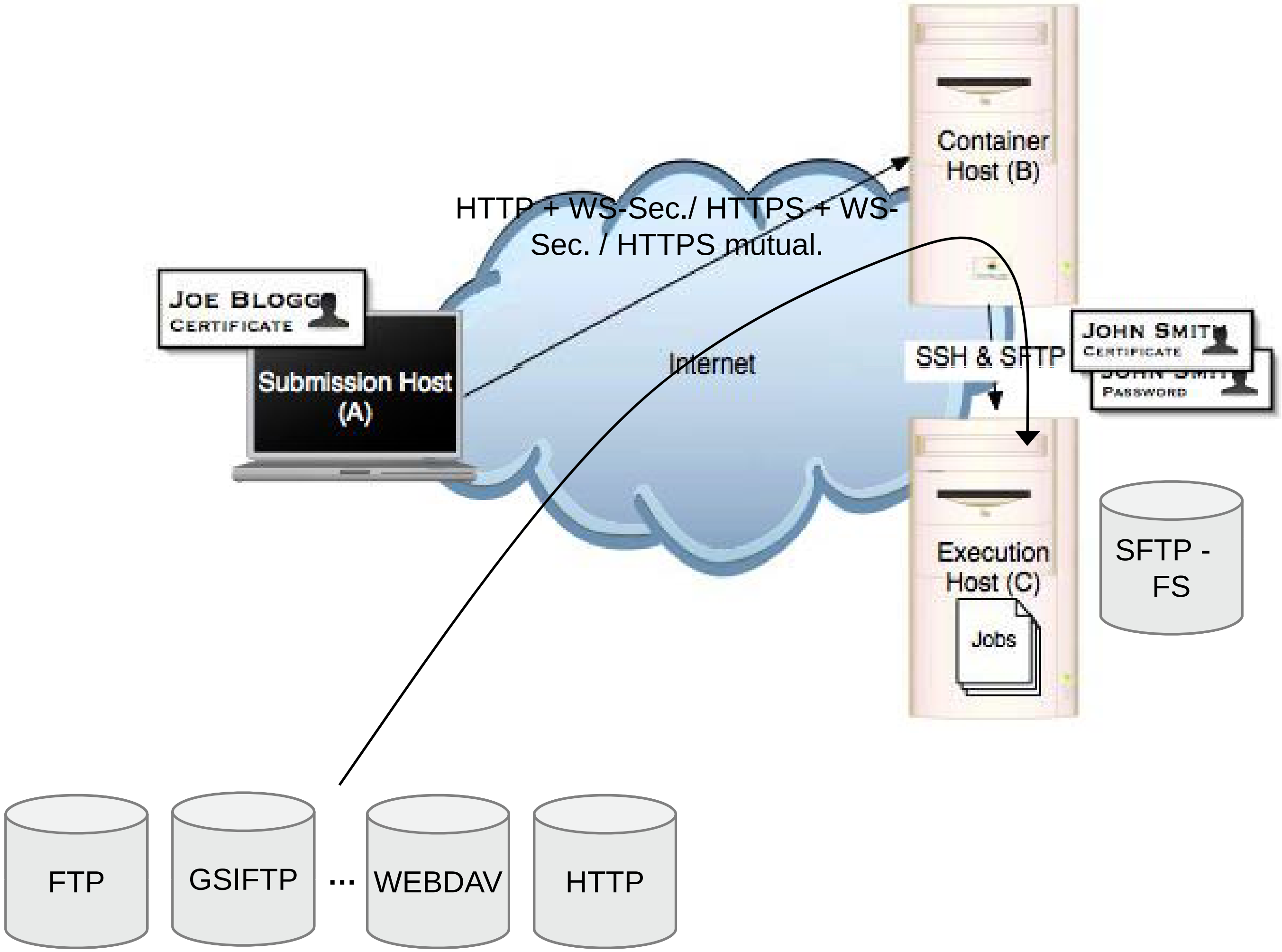
GridSAM is not

- a scheduling service
  - That’s the role of the underlying launching mechanism
  - That’s the role of a super-scheduler that brokers jobs to a set of GridSAM services
- a provisioning service
  - GridSAM runs what’s been told to run
  - GridSAM does not resolve software dependencies and resource requirements

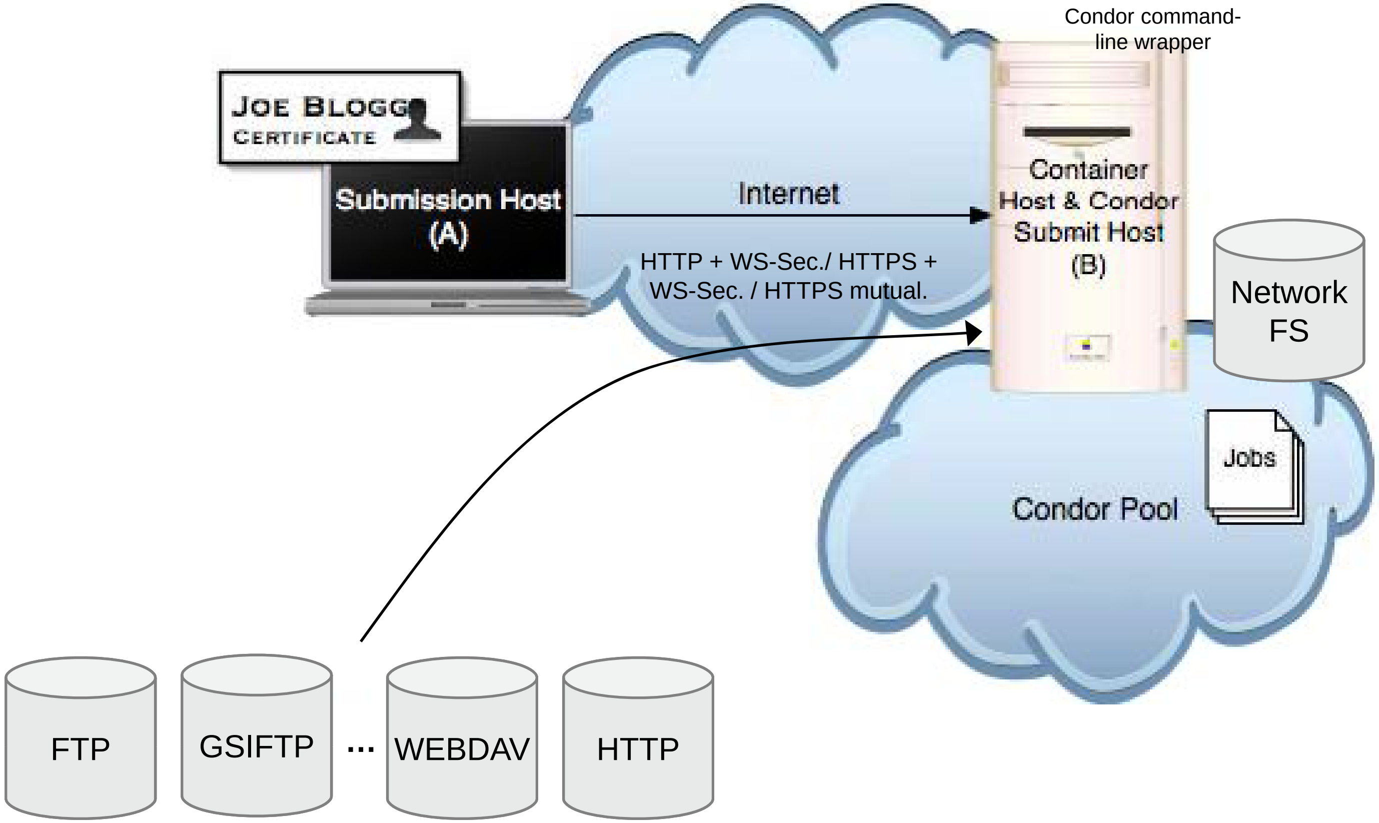
# Deployment Scenario: Forking



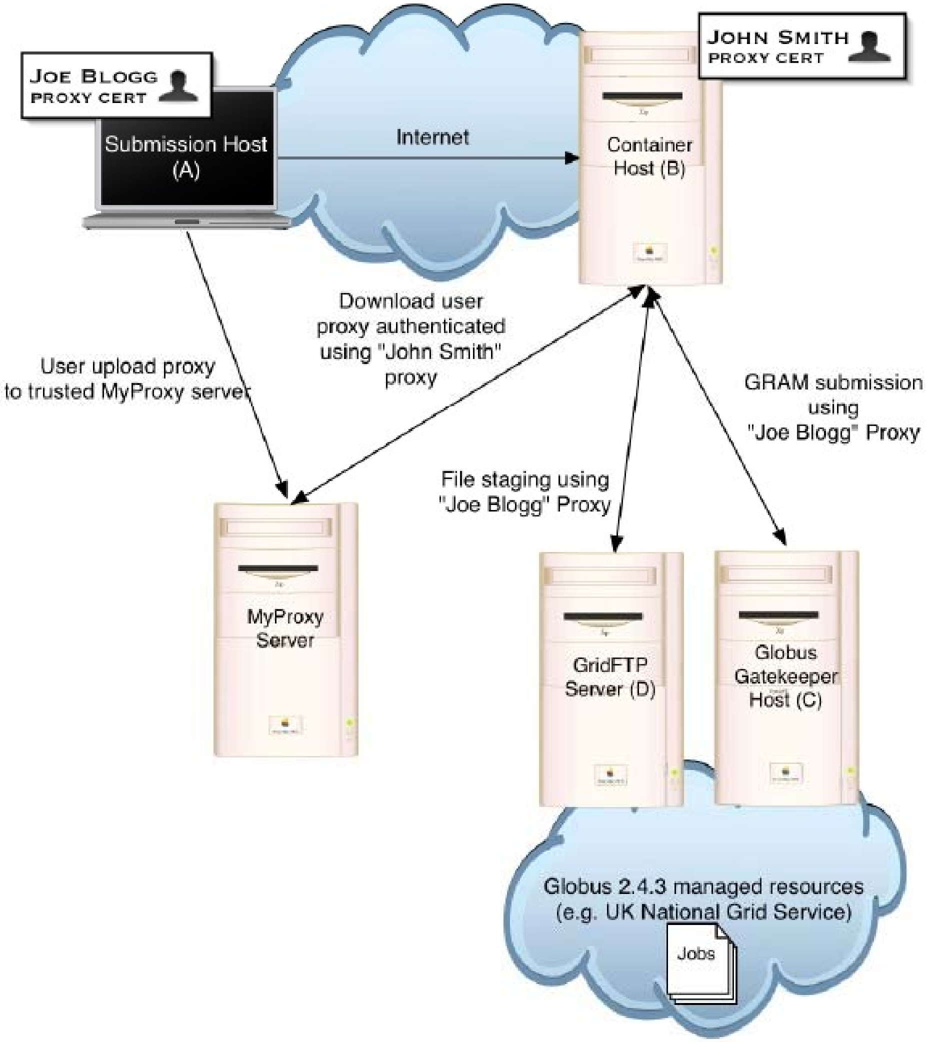
# Deployment Scenario: Secure Shell (SSH)



# Deployment Scenario: Condor Pool


Condor command-line wrapper
Joe Bloggs Certificate
Submission Host (A)
Internet
HTTP + WS-Sec./ HTTPS + WS-Sec. / HTTPS mutual.
Container Host & Condor Submit Host (B)
Network FS
Jobs
Condor Pool
FTP
GSIFTP
...
WEBDAV
HTTP

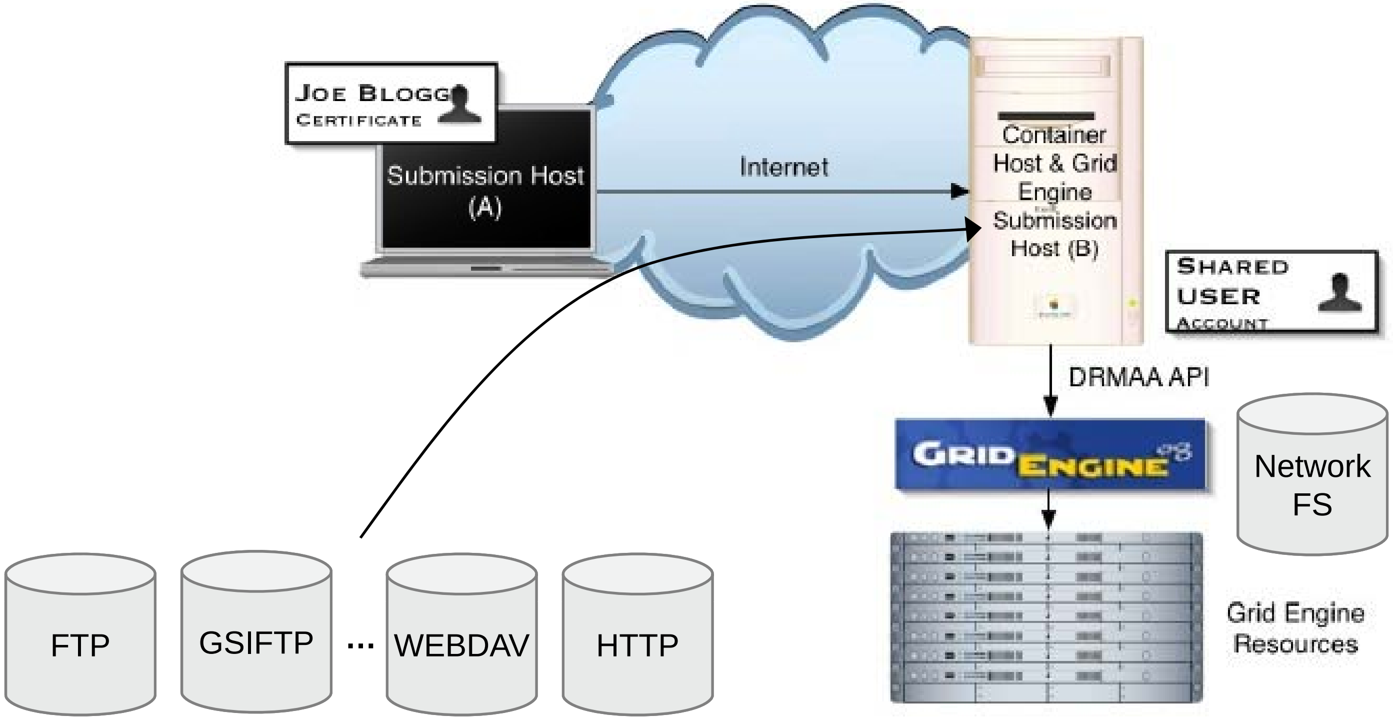
# Deployment Scenario: Globus 2.4.3


Joe Blogg
proxy cert
Submission Host (A)
Internet
John Smith
proxy cert
Container Host (B)
Download user proxy authenticated using "John Smith" proxy
User upload proxy to trusted MyProxy server
GRAM submission using "Joe Blogg" Proxy
File staging using "Joe Blogg" Proxy
MyProxy Server
GridFTP Server (D)
Globus Gatekeeper Host (C)
Globus 2.4.3 managed resources (e.g. UK National Grid Service)
Jobs

London e-Science Centre

# Deployment Scenario: Grid Engine 6

Imperial College London



# Latest Features

Available in v2.0.0-rc1 (released 1/7/06)

- MPI Application through GT2 plugin
  - Simple non-standard JSDL extension `<mpi:MPIApplication/>` that extends `<posix:POSIXApplication/>` with a `<mpi:ProcessorCount/>` element
- Authorisation based on JSDL structure
  - Allow / deny submission based on a set of XPath rules and the identities of the submitter (e.g. distinguished name).
- Prototype Basic Execution Service (ogsa-bes) interface
  - Demonstrated in the mini face-to-face in London last December
  - Shown interoperability with the Uni. Of Virginia BES (.NET based) implementation.

# Upcoming Features

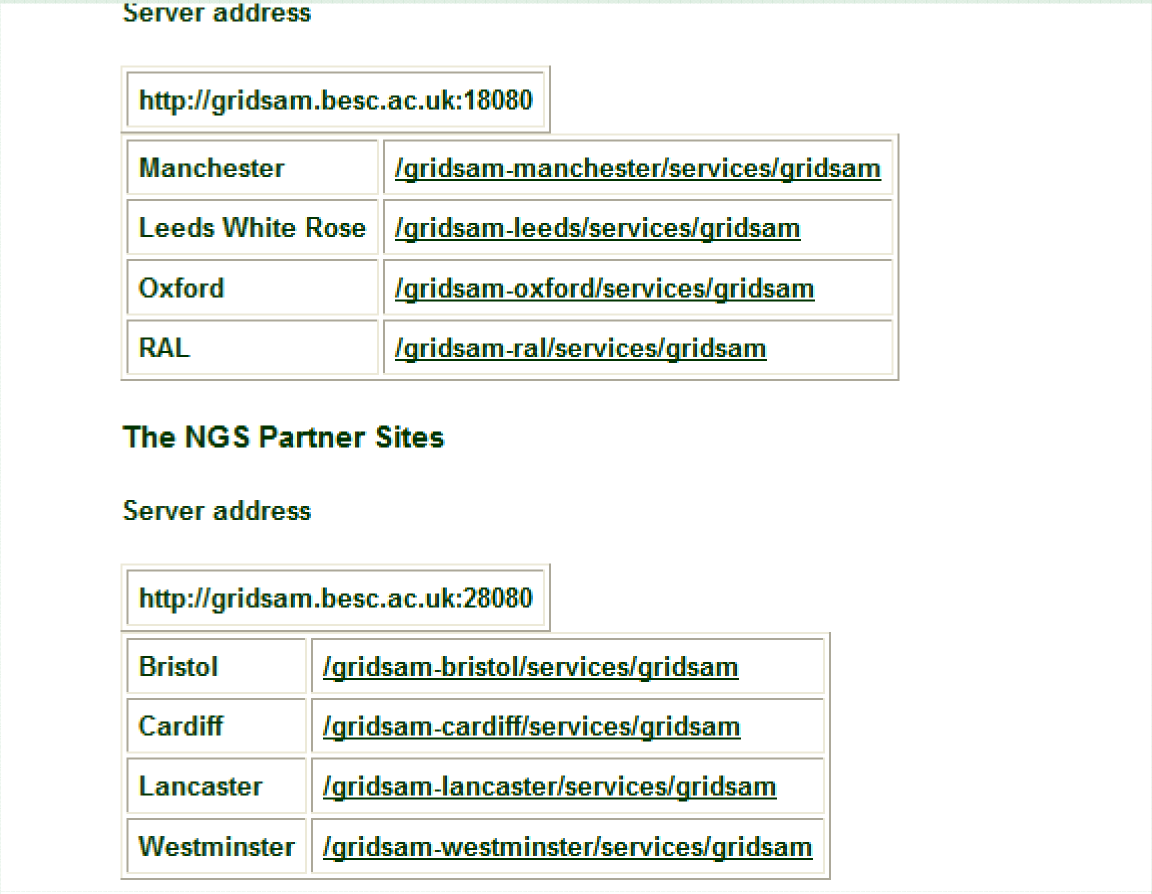
Job State Notification
- Integrate with FINS (WS-Eventing)

Resource Usage Service
- GGF RUS compliant service implementation for recording and querying usages
- Integrate with GridSAM to account for job resource usage

Basic Execution Service
- Continue tracking the changes in the ogsa-bes specification
- Support dual submission WS-interfaces

# Example: GridSAM and the NGS

- Recently deployed by Belfast



Server address

http://gridsam.besc.ac.uk:18080

| | |
|---|---|
| Manchester | /gridsam-manchester/services/gridsam |
| Leeds White Rose | /gridsam-leeds/services/gridsam |
| Oxford | /gridsam-oxford/services/gridsam |
| RAL | /gridsam-ral/services/gridsam |

**The NGS Partner Sites**

Server address

http://gridsam.besc.ac.uk:28080

| | |
|---|---|
| Bristol | /gridsam-bristol/services/gridsam |
| Cardiff | /gridsam-cardiff/services/gridsam |
| Lancaster | /gridsam-lancaster/services/gridsam |
| Westminster | /gridsam-westminster/services/gridsam |

# Requires MyProxy

```
export MYPROXY_SERVER=myproxy.grid-support.ac.uk
myproxy-get-delegation
Enter MyProxy pass phrase:
A credential has been received for user mjm in /tmp/x509up_u24022.
[mjm@tc03 mjm]$ grid-proxy-info
subject  : /C=UK/O=eScience/OU=Edinburgh/L=NeSC/CN=mike
     mineter/CN=proxy/CN=proxy/CN=proxy
issuer   : /C=UK/O=eScience/OU=Edinburgh/L=NeSC/CN=mike
     mineter/CN=proxy/CN=proxy
identity : /C=UK/O=eScience/OU=Edinburgh/L=NeSC/CN=mike mineter
type     : full legacy globus proxy
strength : 1024 bits
path     : /tmp/x509up_u24022
timeleft : 1:53:06
```

# GridSAM on NGS

- To try it:
  - Download client from OMII-UK
  - Upload proxy to MyProxy server (see later today)
- AND
  - Delete any log files – your passphrase is in them
- Today:
  - Not using GridSAM directly – but it is used by AHE, Application Hosting Environment.

NGS
National Grid Service

# Summary

- JSDL: standard, extendable language for describing jobs
- Used in GridSAM and in OGSA-BES (Basic Execution Service) web services
- Can build higher level tools for job execution on diverse resources
- Example of JSDL use today – NGS Applications Repository
- Example of GridSAM use today – Application Hosting Environment

# Further Information

- Official Download
  - – http://www.omii.ac.uk
- Project Information and Documentation
  - – http://gridsam.sourceforge.net