EPCC Sun Data and Compute Grids
Project Update

Using Sun Grid Engine and Globus for Multi-Site Resource Sharing

Grid Engine Workshop, Regensburg, September 2003

Geoff Cawood
Edinburgh Parallel Computing Centre (EPCC)

Email: geoffc@epcc.ed.ac.uk

Overview

- Background
- Project aims
- Current status
- Software deliverables
  - TOG (Transfer-queue Over Globus)
    - Use in ODD-Genes project
  - JOSH (JOb Scheduling Hierarchically)
- Conclusions

Background

EPCC

- Edinburgh Parallel Computing Centre: www.epcc.ed.ac.uk
- Part of the University of Edinburgh in Scotland
- “A technology-transfer centre for high-performance-computing”
- Representing NeSc (National e-Science Centre)

A collaborative project between EPCC and Sun

- Referred to as ‘Sun DCG' or ‘Sungrid’ within EPCC
- Project website: www.epcc.ed.ac.uk/sungrid
- Team of 4 people - Thomas Seed also here
- 57 person months of EPCC effort over 2 years
- Sun approve project deliverables
- Funded by UK DTI/EPSRC e-Science core program

Project Goal

Final project goal

- "Develop a fully Globus-enabled compute and data scheduler based around Grid Engine, Globus and a wide variety of data technologies"

What does that mean in practice?

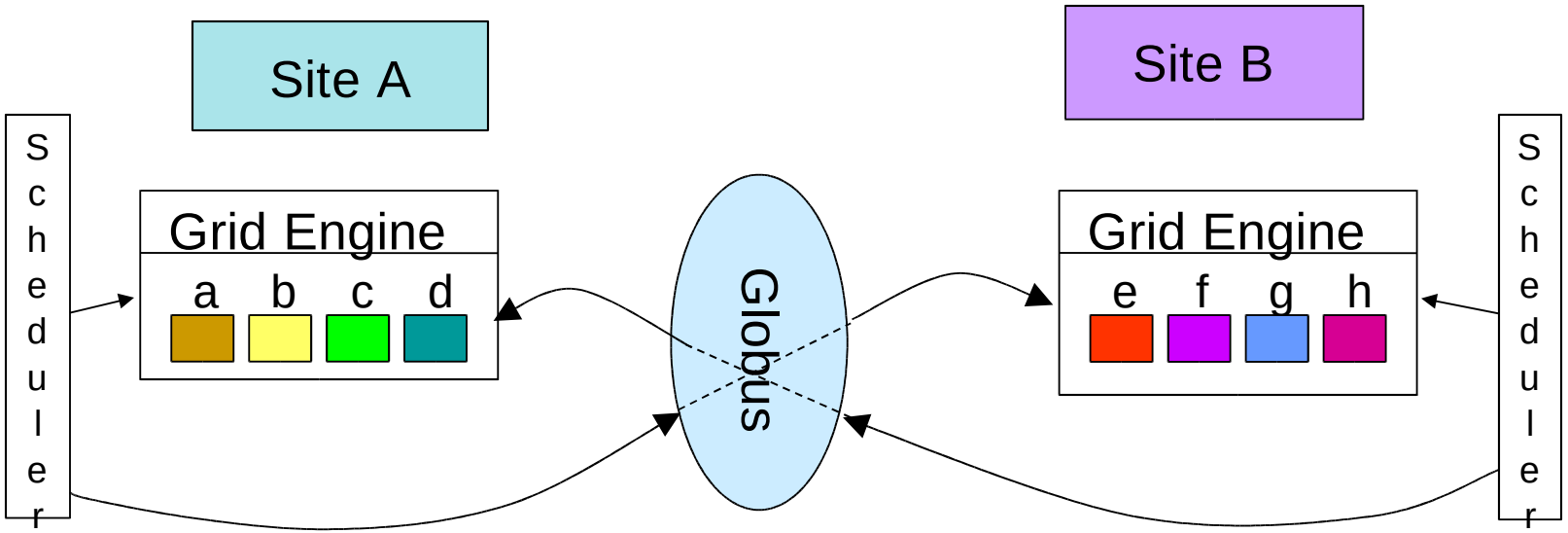
Identify five key functional aims

- 1. Job scheduling across Globus to remote Grid Engines
- 2. File transfer between local client site and remote jobs
- 3. File transfer between *any* site and remote jobs
- 4. Allow 'datagrid aware' jobs to work remotely
- 5. Data-aware job scheduling

Functional Aim

1. Job scheduling across Globus to remote GEs

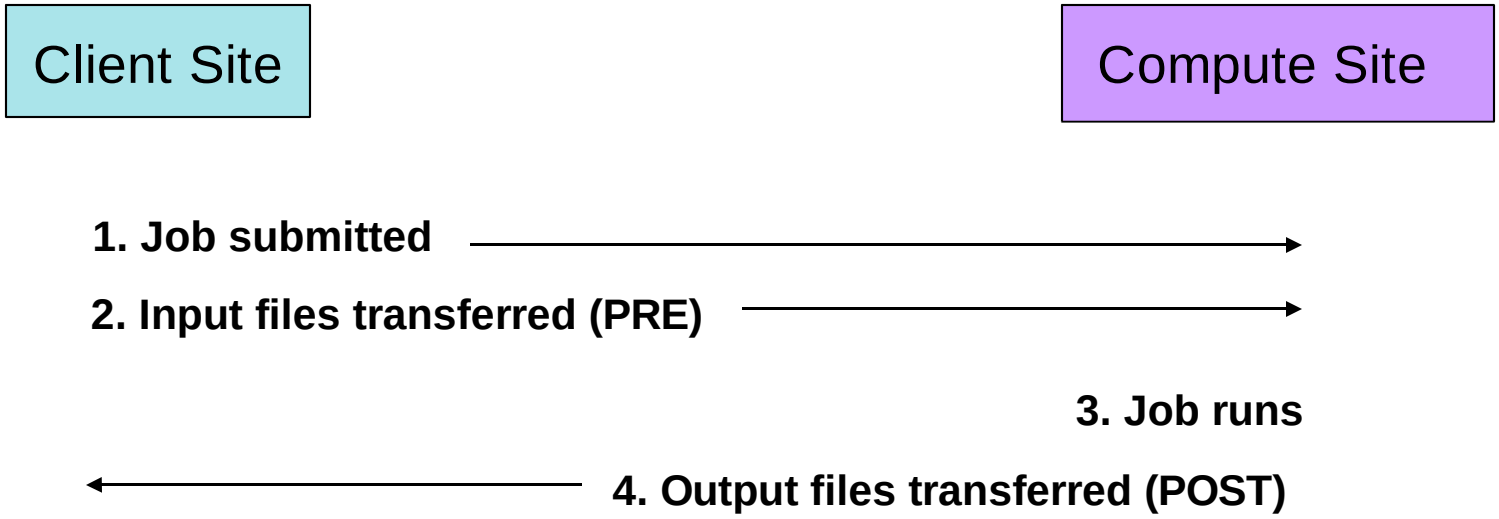
- Schedule jobs securely onto remote machines
- Allow collaborating enterprises to share compute resources
- Efficiency benefits of using lightly loaded remote machines



Functional Aim

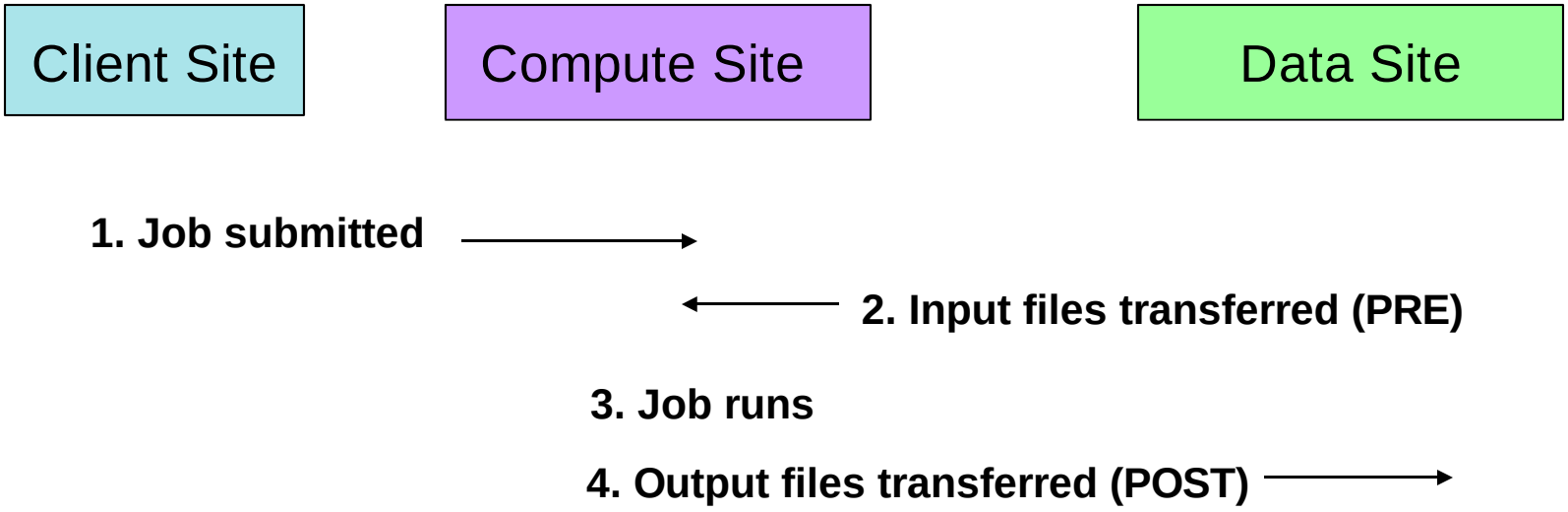
2. File transfer between local site and remote jobs

- Data staging, executable staging
- Allow jobs expecting local file I/O to function properly



3. File transfer between *any* site and remote jobs

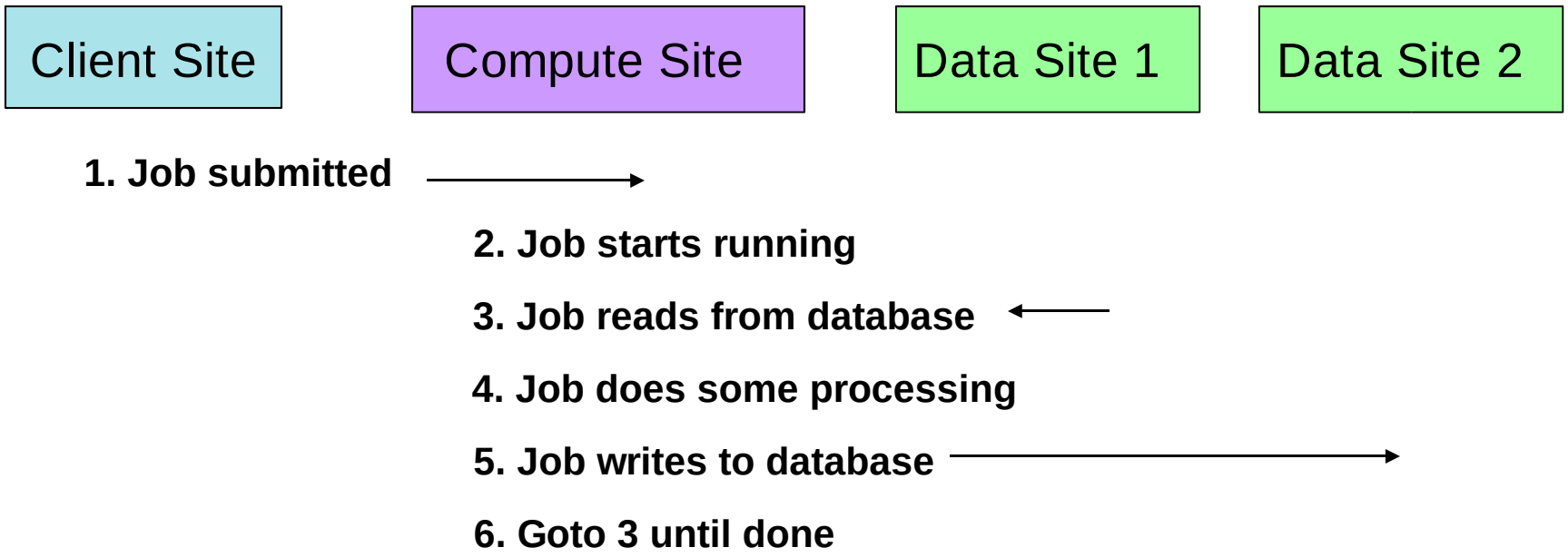
- Allow access to shared data files anywhere
  - HTTP sites, FTP sites, GridFTP sites



Functional Aim

4. Allow 'datagrid aware' jobs to work remotely

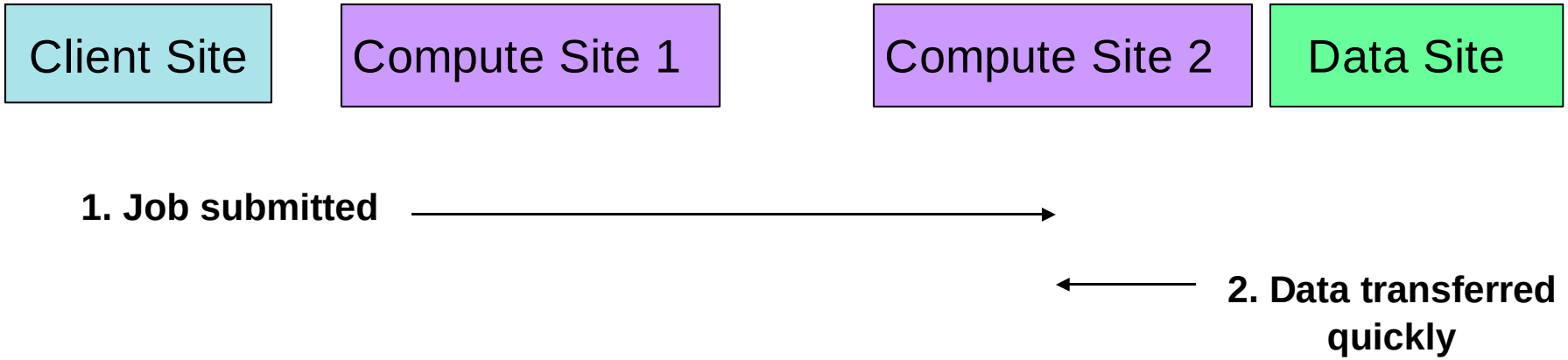
- Ensure that attempts by a job to access Globus-enabled data sources are not hindered by running the job remotely
  - Different style of job - dynamic data access not just pre and post
  - Data sources: GridFTP sites, OGSA-DAI databases, SRB



Functional Aim

5. Data-aware job scheduling

- Schedule data-intensive jobs ‘close’ to the data they require
- In this example Compute Site 2 has better access to the Data Site



Project Status

Early workpackages

- Investigations, self-education
- Delivering documents

WP 1: Analysis of existing Grid components (finished)

- WP 1.1: UML analysis of core Globus 2.0
- WP 1.2: UML analysis of Grid Engine
- WP 1.3: UML analysis of other Globus 2.0
  - Documents available at project web site
- WP 1.4: Globus 3.0 Investigation
- WP 1.5: Exploration of data technologies
  - Documents will be available after approval by Sun

WP 2: Requirements Capture & Analysis (finished)

- Documents available at project web site

Project Status

Later workpackages

- Design, development and test
- Delivering design documents and software

WP 3: Prototype Development (finished)

- **TOG** (**T**ransfer-queue **O**ver **G**lobus) software produced
  - Software and docs available from Grid Engine community web site
  - http://gridengine.sunsource.net/project/gridengine/tog.html

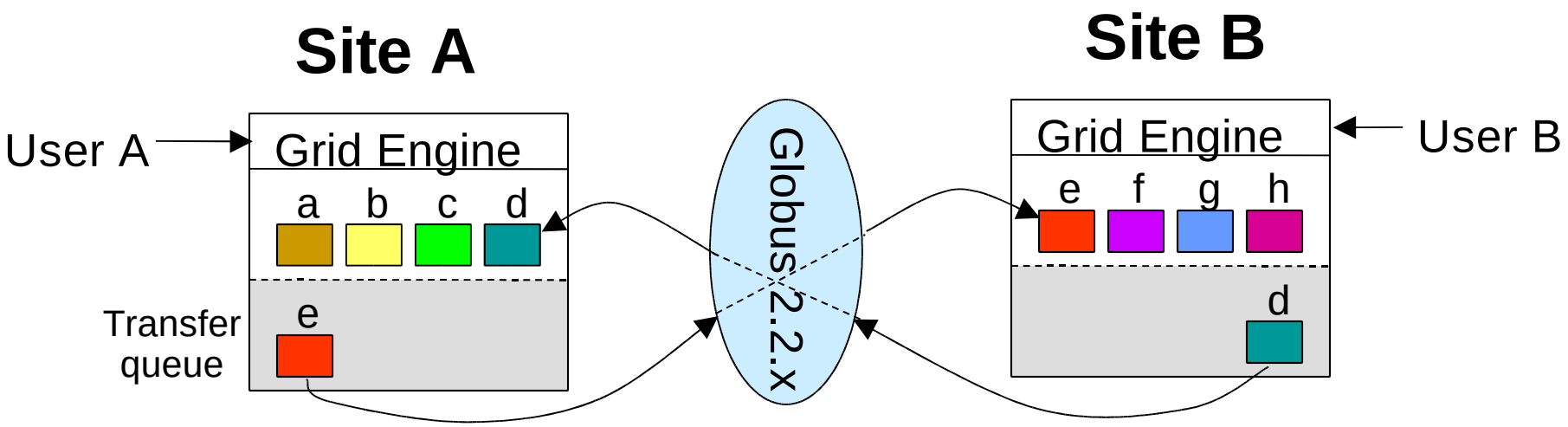
WP 4: Hierarchical Scheduler Design (finished)

- **JOSH** (**JO**b Scheduling **H**ierarchically) software designed
  - Documents will be available after approval by Sun

WP 5: Hierarchical Scheduler Development

- Starting September 2003
- Finishing end January 2004

TOG (Transfer-queue Over Globus)



- No new meta-scheduler - solution uses Grid Engine at two levels
- Integrates GE and Globus 2.2.x
- Supply GE execution methods (starter method etc.) to implement a 'transfer queue' which sends jobs over Globus to a remote GE
- GE complexes used for configuration
- Globus GSI for security, GRAM for interaction with remote GE
- GASS for small data transfer, GridFTP for large datasets
- Written in Java - Globus functionality accessed through Java COG kit

TOG Software

Functionality

- 1. Job scheduling across Globus to remote Grid Engines
- 2. File transfer between local client site and remote jobs
  - Add special comments to job script to specify set of files to transfer between local site and remote site
- 4. Allow 'datagrid aware' jobs to work remotely
  - Use of Globus GRAM ensures proxy certificate is present in remote environment

Absent

- 3. File transfer between *any* site and remote jobs
  - Files are transferred between remote site and local site only
- 5. Data-aware job scheduling

TOG Software

Pros

- Simple approach
- Usability
  - Existing Grid Engine interface
  - Users only need to learn Globus certificates
- Remote administrators still have full control over their resources

Cons

- Low quality scheduling decisions
  - State of remote resource – is it fully loaded?
  - Ignores data transfer costs
- Scales poorly - one local transfer queue for each remote queue
- Manual set-up
  - Configuring the transfer queue with same properties as remote queue
- Java virtual machine invocation per job submission

TOG in Action

ODD-Genes Project

- Uses SunDCG and OGSA-DAI to demonstrate a scientific use for the grid (bioinformatics)

Shown at UK All Hands Meeting 2003 in Sept

Will be shown at Supercomputing 2003 in Nov

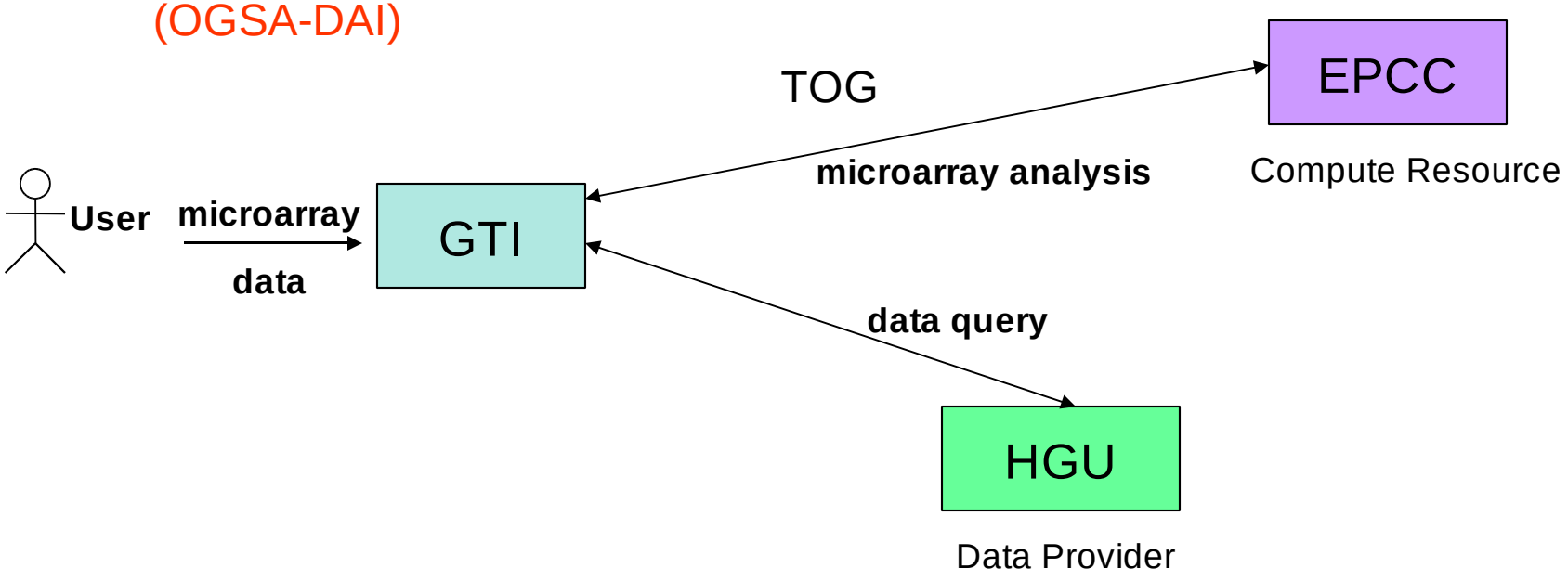
Demo links 3 sites within Edinburgh University

- Scottish Centre for Genomic Technology and Informatics (GTI)
- Medical Research Council's Human Genetics Unit in Edinburgh (HGU)
- One of EPCC's high performance compute resource (Sun Fire 15K)

TOG used at GTI to access EPCC compute resource

Overview of ODD-Genes Project

- User submits microarray data to GTI
- GTI uses EPCC to perform analysis (TOG)
- User views analysis results (gene expressions)
- User queries for more information on genes from HGU (OGSA-DAI)



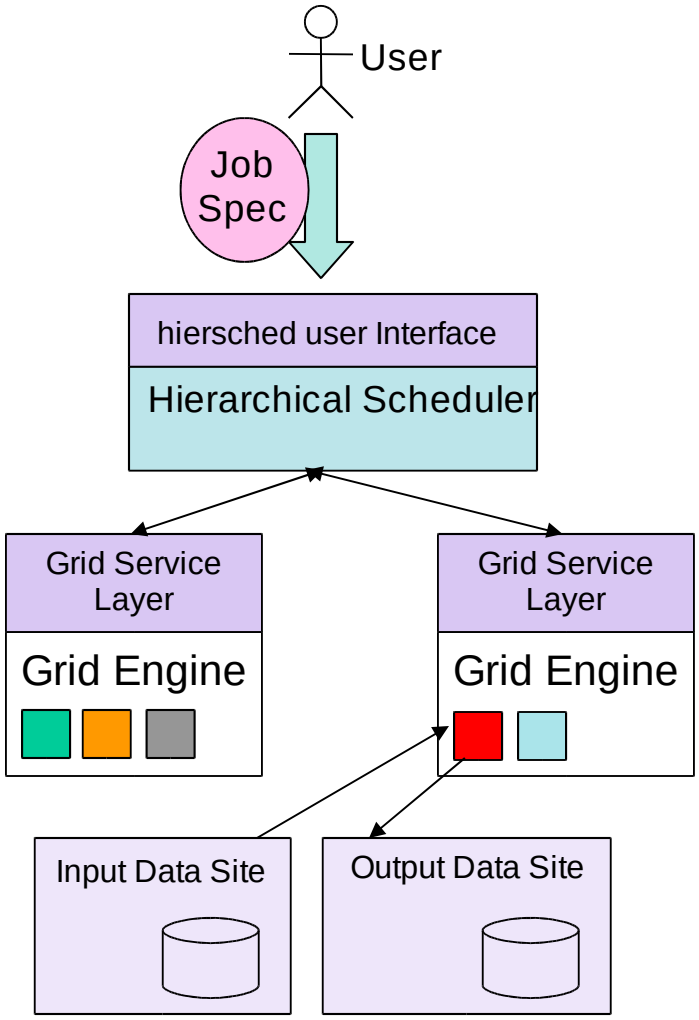
JOSH (JOb Scheduling Hierarchically)

Developing JOSH software

- Address the shortcomings of TOG
- Incorporate Globus 3 and grid services

Adds a new 'hierarchical' scheduler above Grid Engine

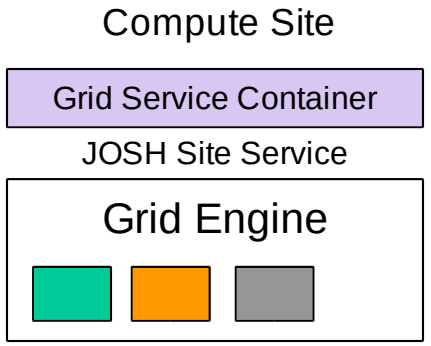
- Command line interface
- `hiersched submit_ge`
  - Takes GE job script as input (embellished with data requirements)
  - Queries grid services at each compute site to find best match and submits job
  - Job controlled through resulting 'job locator'



JOSH Sites

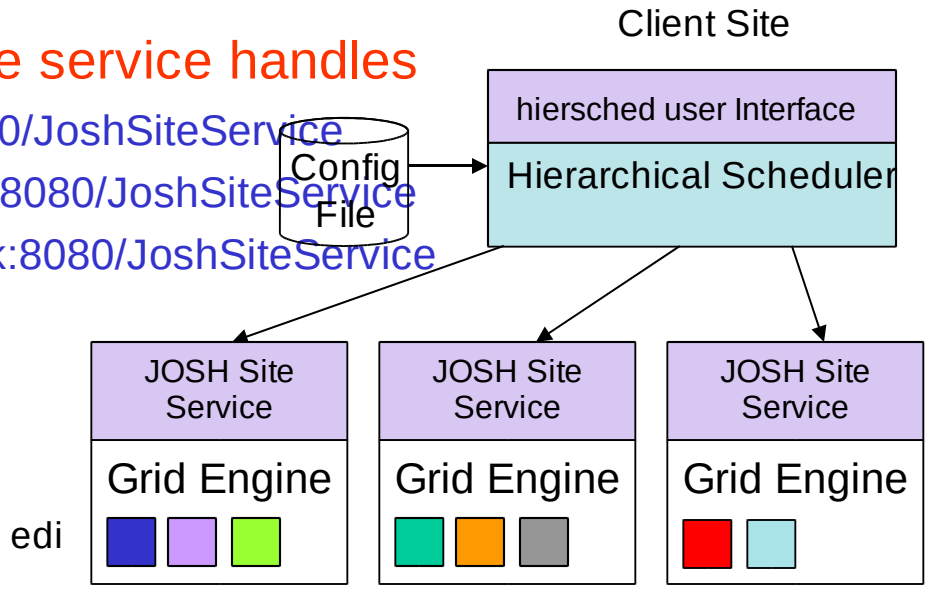
Compute sites

- One Grid Engine per compute site
- A persistent JOSH 'site service' runs in a Grid Service Container at each compute site



Configuration file at client site

- Maps compute site names to site service handles
  - 'edi' -> http://www.rush.ed.ac.uk:8080/JoshSiteService
  - 'glasgow' -> http://www.otf.gla.ac.uk:8080/JoshSiteService
  - 'london' -> http://www.balrog.ln.ac.uk:8080/JoshSiteService
- Defines a 'pool' of candidate sites for the hierarchical scheduler to choose from



JOSH Usage

```
geoffc% hiersched sitestatus

Hierarchical Scheduler Site Status
==================================

Site Name Status Site Handle
--------- ------ -----------
edi       Up     http://www.rush.ed.ac.uk:8080/JoshSiteService
glasgow   Down   http://www.otf.gla.ac.uk:8080/JoshSiteService
london    Up     http://www.balrog.ln.ac.uk:8080/JoshSiteService

geoffc% hiersched submit-ge myjob.sh
edi:4354
geoffc% hiersched jobstatus edi:4354
Pending
geoffc% hiersched terminate edi:4354
SUCCESS
geoffc%
```

Scheduling Algorithm

The hierarchical scheduler chooses a site for a job according to the following criteria:

- Physical capability
  - Sites which have no queues that can satisfy the job are rejected
- Load score
  - Define a site's load score as the minimum load of its capable queues for a given job
  - Sites with lower load are favoured
- Data proximity
  - Sites 'closer' to their data sources are favoured

Weighting factors

- Can supply multipliers for load and data proximity with the job
  - Eg. so data proximity can be ignored for compute intensive jobs

JOSH Implementation

One hiersched job submission is implemented as *three* Grid Engine jobs at the compute site

- 1. PRE Job - pulls input data files to the compute site
- 2. MAIN Job - the user's original job script
- 3. POST Job - pushes output data files to their destinations and cleans up

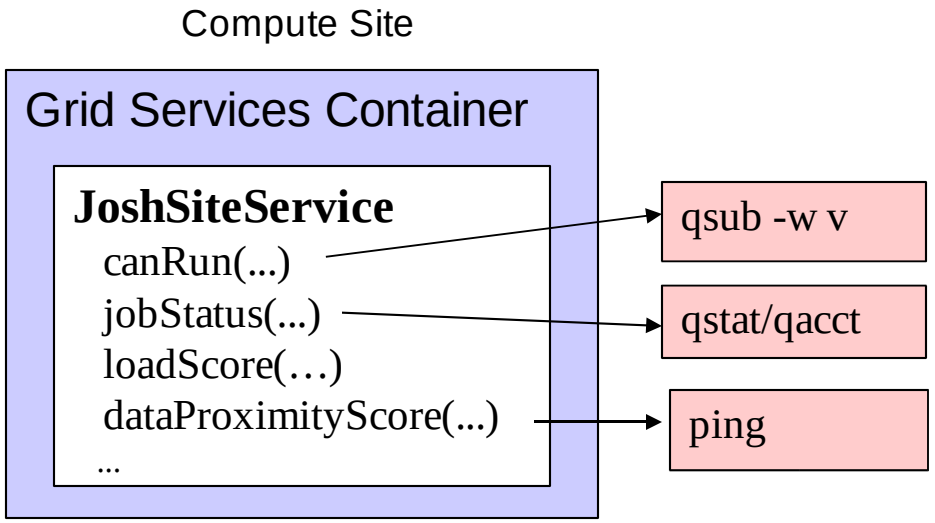
Separate jobs approach solve various issues

- Don't want a high performance queue clogged up with a big data transfer task
  - PRE and POST jobs can be run in dedicated data transfer queue
- Do want a terminated job to be cleaned up and partial output data returned
  - Terminating MAIN job releases POST job

Compute Site Implementation

Grid service operations

- Call out to GE (and other) executables
- Parse output (fiddly)
- Some more-specific GE commands would be handy



Globus 3 operations always run as container owner

- But job submission, termination etc. must run under the client user's remote account
- Ensure correct privileges, prevent killing someone else's job etc.
- Workround is to use the GT3 Managed Job Service - bit ugly
- Allows a *script* to run under the client user's account

Pros

- Satisfies the five functionality goals
- Need only minor additions to existing GE job scripts
  - Data requirement comments
- Remote administrators still have full control over their GEs
- Tries to make use of existing GE functionality eg. 'can run'

Cons

- Latency in decision making
- Not so much 'scheduling' as 'choosing'
- Grid Engine specific solution

Conclusions

Significant progress since last GE Workshop

- Learned a lot
- Focussed on achievable goal that will hopefully have some wider use
- Released various documents and the TOG software

Prototyping exercise proved valuable

- Developing TOG was a worthwhile and educational first step
- Already being used for ODD-Genes project, potentially others

Hierarchical scheduler implementation starting

- JOSH software should address the main TOG limitations
- Early adoption of Globus 3 is still a bit of a risk

Our interests at this GE Workshop

- GE developments, Globus 3 experiences, multi-site scheduling
- Anything to ease our final development phase!