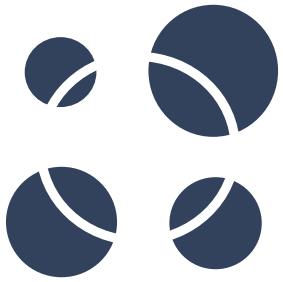
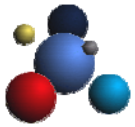
IBERIAN GRID INFRASTRUCTURE CONFERENCE
Santiago de Compostela, Spain
May 15, 2007



Grid Scheduling Architectures

Ignacio M. Llorente

Distributed Systems Architecture Group
Universidad Complutense de Madrid
http://asds.dacya.ucm.es

Contents

1. **Computing Resources**
2. Grid Middleware
3. A Taxonomy for Grid Computing Infrastructures
4. A Note on Usability

1. Computing Resources

1.1. Parallel and Distributed Computing

Goal of Parallel and Distributed Computing

- ***Efficient*** execution of computational or data-intensive applications

Types of Computing Environments

High Performance Computing (HPC) Environments

- Reduce the execution time of a single distributed or shared memory parallel application (MPI, PVM, HPF, OpenMP…)
- Performance measured in floating point operations per second
- Sample areas: CFD, climate modeling…

High Throughput Computing (HTC) Environments

- Improve the number of executions per unit time
- Performance measured in number of jobs per second
- Sample areas: HEP, Bioinformatics, Financial models…

1. Computing Resources

1.2. Types of Computing Platforms

Centralized Coupled

- **Network Links**
- **Administration**
- **Homogeneity**

Decentralized Decoupled

SMP (Symmetric Multi-processors)

MPP (Massive Parallel Processors)

Clusters

Network Systems Intranet/Internet

High Performance Computing

High Throughput Computing

1. Computing Resources

1.3. Local Resource Management Systems

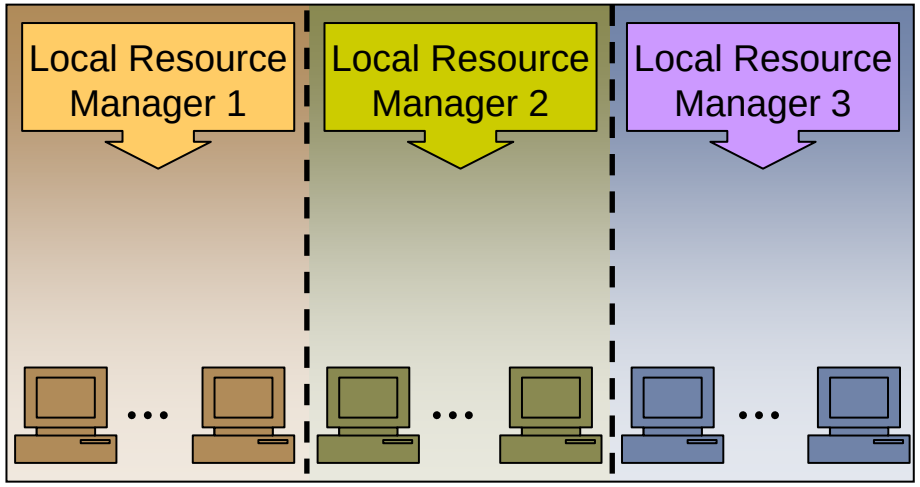
Management of Computing Platforms

- Computing platforms are managed by **Local Resource Management (LRM) Systems**
  1. Batch queuing systems for HPC servers
  2. Resource management systems for dedicated clusters
  3. Workload management systems for network systems
- There aim is to maximize the system *performance*

Independent Suppliers	*Open Source*	*OEM Proprietary*
2 3 *Platform Computing* **LSF**	2 *Altair* **Open PBS**	1 *IBM* **Load Leveler**
2 *Altair* **PBS Pro**	2 3 *University of Wisconsin* **Condor**	1 *Cray* **NQE**
	2 3 *Sun Microsystems* **SGE**	

LRM Systems Limitations

- Do not provide a common interface or security framework
- Based on proprietary protocols
- **Non-interoperable computing vertical silos** within a single organization
  - Requires specialized administration skills
  - Increases operational costs
  - Generates over-provisioning and global load unbalance



Only a small fraction of the infrastructure is available to the user

Infrastructure is fragmented in non-interoperable computational silos

Contents

1. Computing Resources
2. **Grid Middleware**
3. A Taxonomy for Grid Computing Infrastructures
4. A Note on Usability

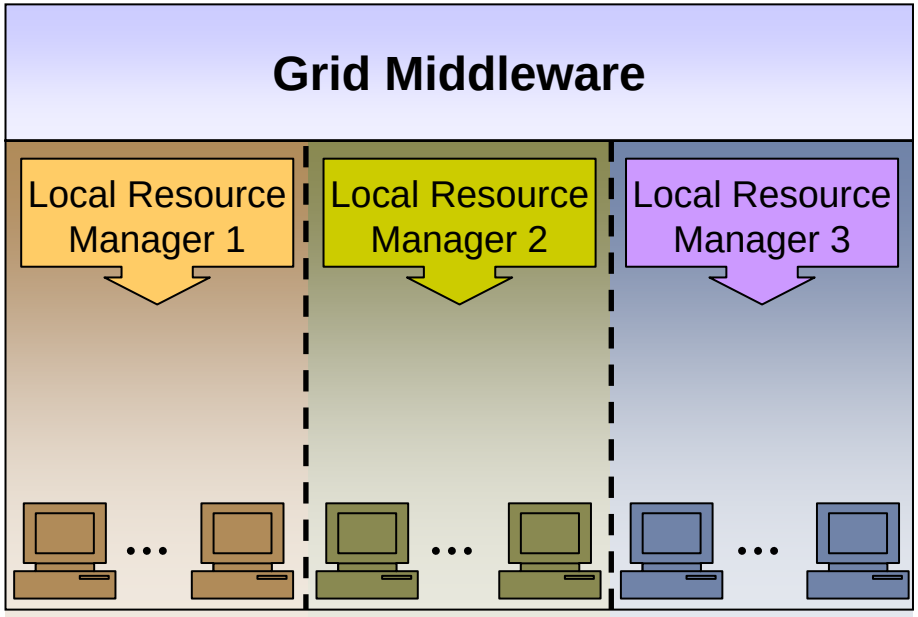
2. Grid Middleware

2.1. Integration of Different Administrative Domains

"Any problem in computer science can be solved with another layer of indirection… *But that usually will create another problem*." David Wheeler

A New Abstraction Level

"A (*computational*) grid offers a common layer to integrate heterogeneous computational platforms (vertical silos) and/or administrative domains by defining a consistent set of abstraction and interfaces for access to, and management of, shared resources"



Common Interface: User can access a wide set (number and type) of resources.

Infrastructure: Computational and storage resources, network and LRM Systems

2. Grid Middleware

2.1. Integration of Different Administrative Domains

Grid Middleware (a computational view)

- **Services in the Grid Middleware layer**
  - Security
  - Information & Monitoring
  - Data Management
  - Execution
  - Meta-scheduling
- **Open Source Middleware Distributions**

glite.web.cern.ch

www.unicore.org

www.omii.ac.uk

www.gria.org

vdt.cs.wisc.edu

- **Open Source Middleware Communities**

The Globus Alliance (dev.globus.org)

Why Globus?…

- **Open Community Project** based on Apache Jakarta model:
  - Control of each individual project is in hand of the committers
  - Public development infrastructure for each project: CVS, bugzilla, mailing list, and Wiki
  - Each project goes through an incubation process before becoming a Globus project.
- The **Globus Toolkit (GT)** distribution integrates a selected group of Globus technologies
- GT **provides basic services** to allow secure remote operation over multiple administration domains with different LRM systems and access policies.

2. Grid Middleware

2.2. The Globus Toolkit

Globus Components

Globus Projects

GridWay

Globus Incubator Projects

Gridshib

DDM

LRMA

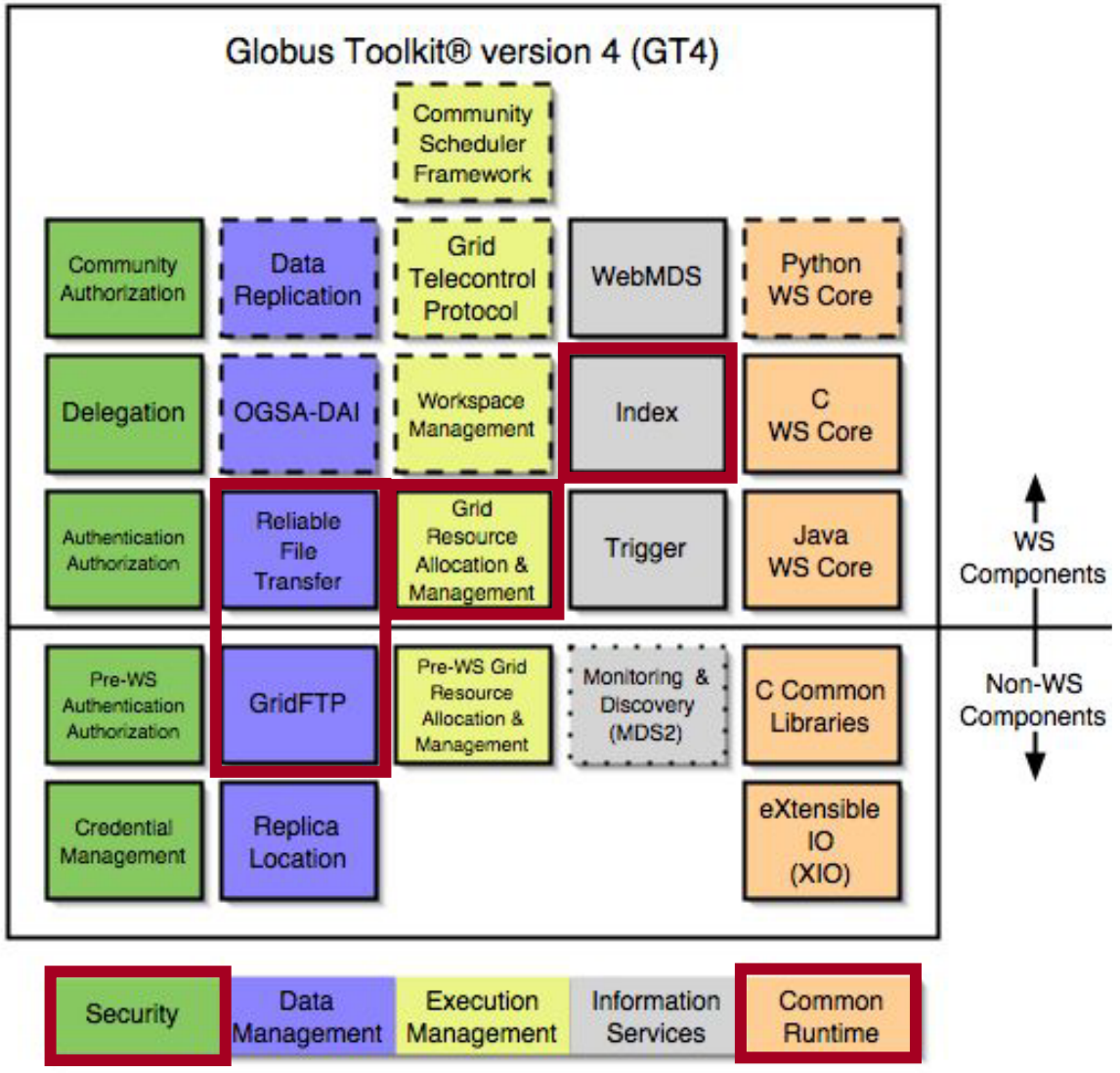
GRAADS

CoG Workflow

■ ■ ■

and many more!

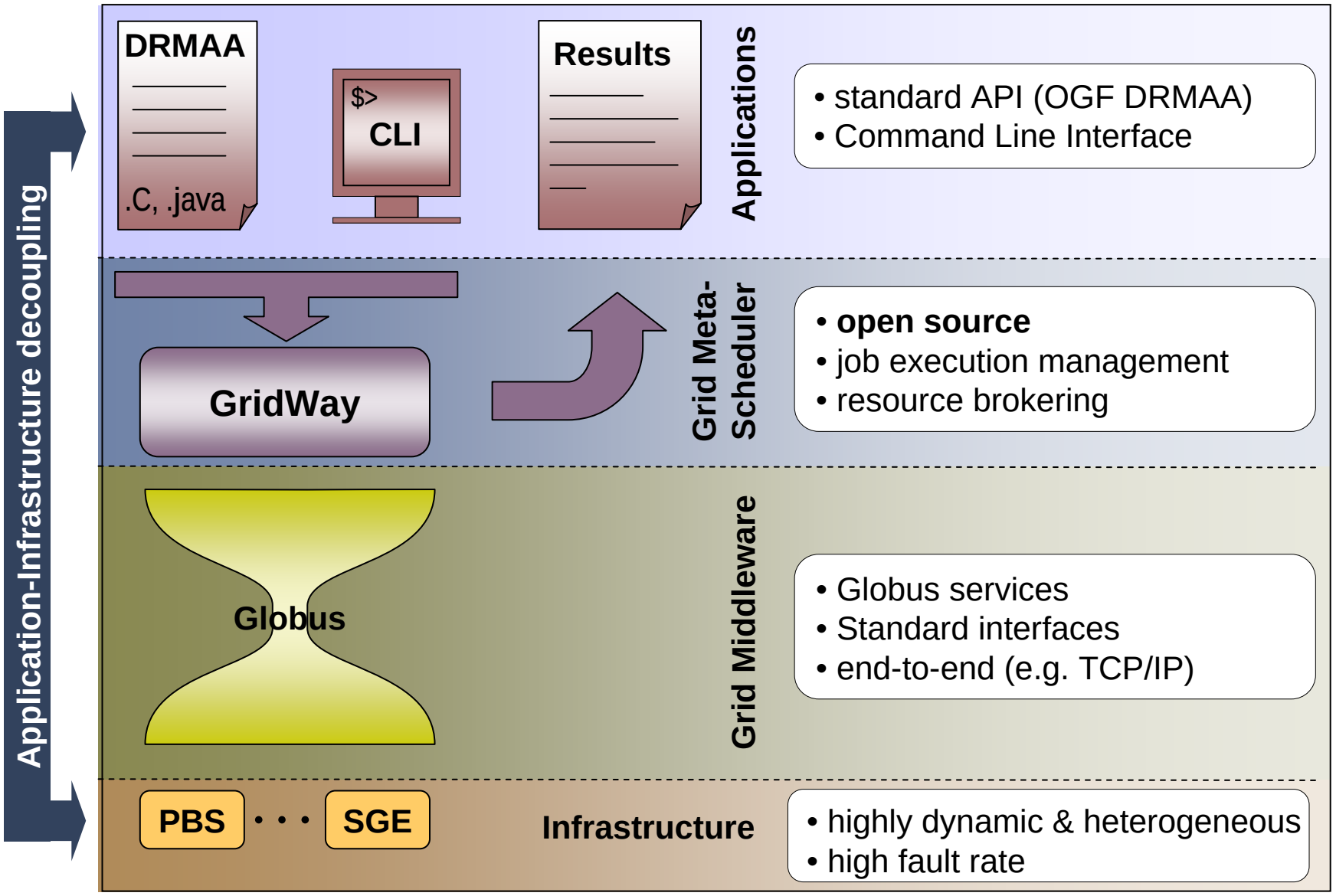
Min. components for a Computational Grid



2. Grid Middleware

2.3. The GridWay Meta-scheduler

Architecture of a Computational Grid



Benefits

Integration of non-interoperable computational platforms (Organization)

- Establishment of a uniform and flexible infrastructure
- Achievement of greater utilization of resources and higher application throughput

Support for the existing platforms and LRM Systems (Sys. Admin.)

- Allocation of grid resources according to management specified policies
- Analysis of trends in resource usage
- Monitoring of user behavior

Familiar CLI and standard APIs (End Users & Developers)

- High Throughput Computing Applications
- Workflows

Features

Workload Management

- Advanced (Grid-specific) scheduling policies
- Fault detection & recovery
- Accounting
- Array jobs and DAG workflows

User Interface

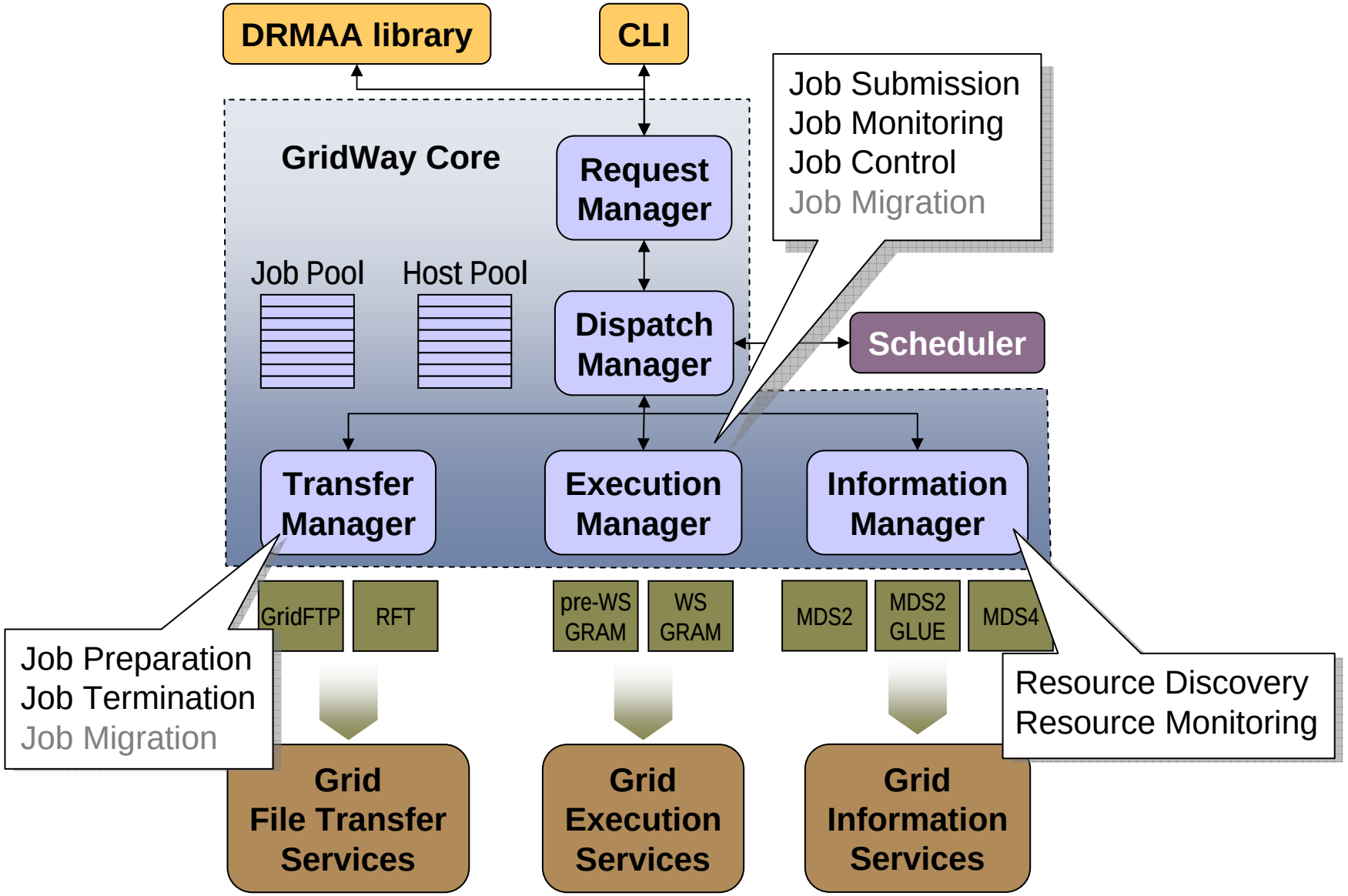
- OGF standards: JSDL & DRMAA (C and JAVA)
- Analysis of trends in resource usage
- Command line interface, similar to that found on local LRM Systems

Integration

- Straightforward deployment as new services are not required
- interoperability between different infrastructures

2. Grid Middleware

2.3. The GridWay Meta-scheduler



2. Grid Middleware

2.4. Grid Computing Infrastructures

Centralized Coupled

- **Network Links**
- **Administration**
- **Homogeneity**

Decentralized Decoupled

SMP (Symmetric Multi-processors)	**MPP** (Massive Parallel Processors)	**Clusters**	**Network Systems Intranet/Internet**	**Grid Infrastructures**



High Performance Computing

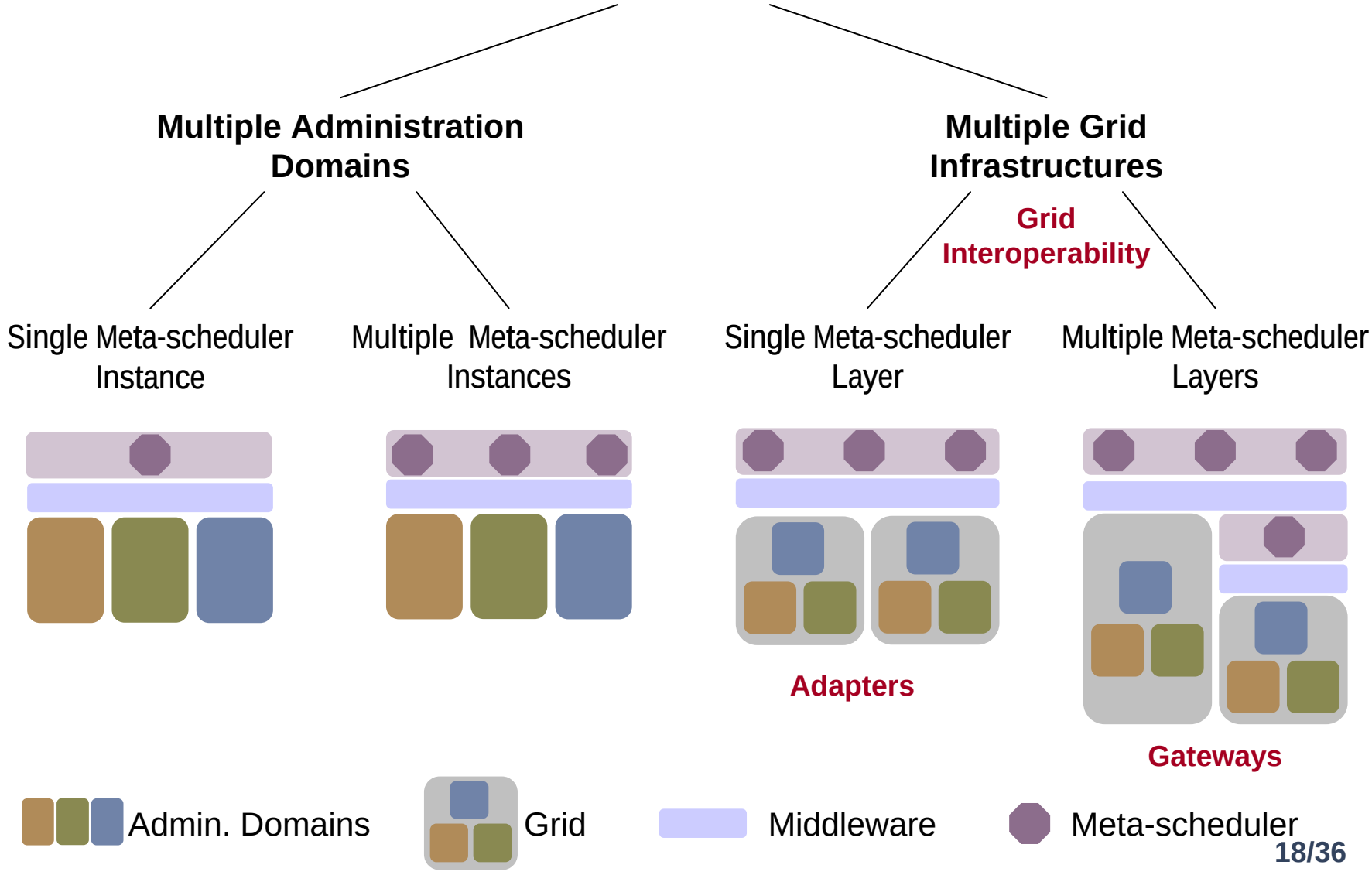
High Throughput Computing

Contents

1. Computing Resources
2. Grid Middleware
3. **A Taxonomy for Grid Computing Infrastructures**
4. A Note on Usability

3. A Taxonomy for Grid Computing Infrastructures

Taxonomy



Single Meta-Scheduler Grids

Characteristics

- One meta-scheduler instance with access to multiple administration domains
- Small scale infrastructures (campus or enterprise)
- Can be geographically distributed in different sites

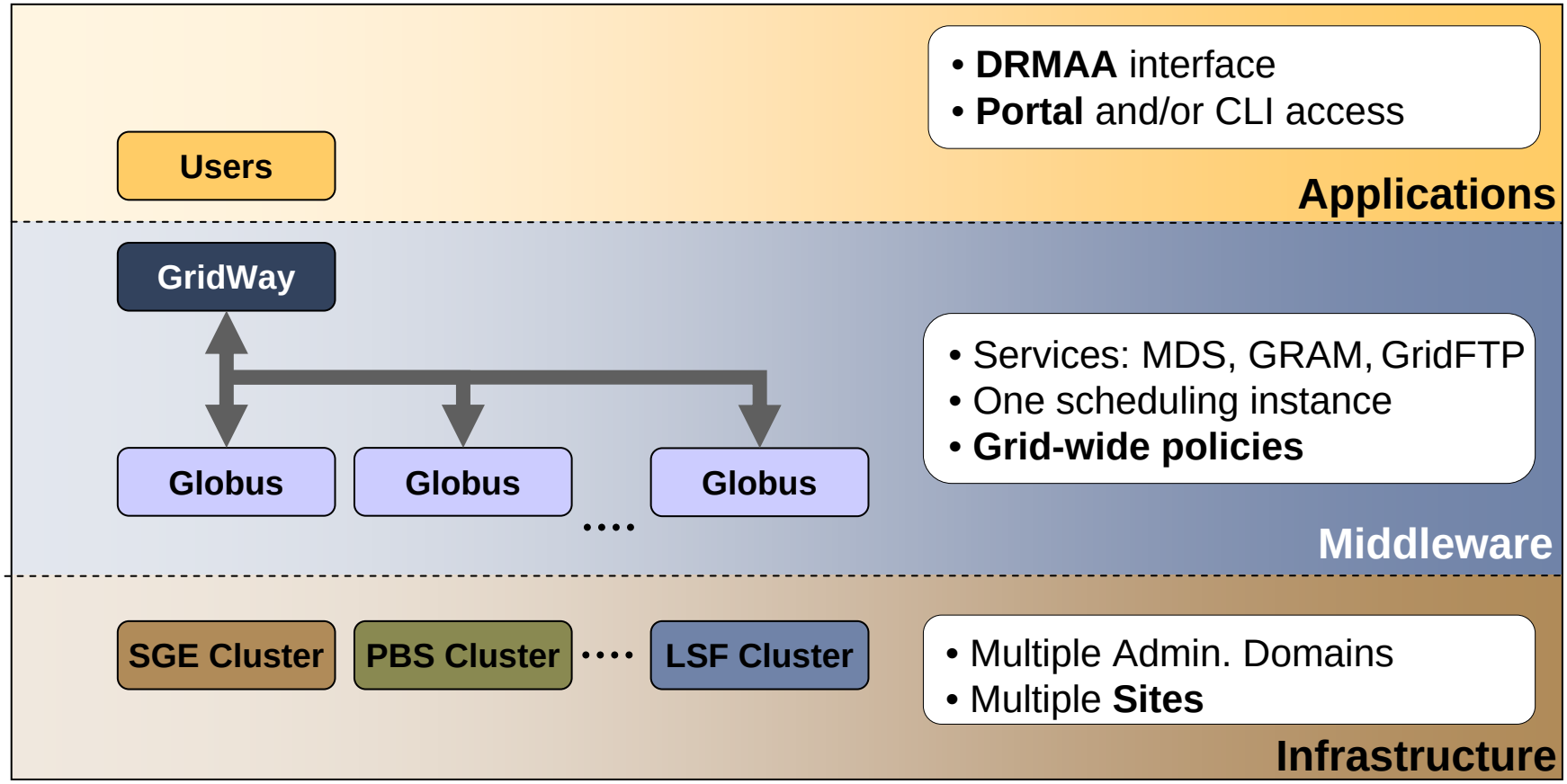
Goal & Benefits

- Integrate multiple Admin. Domains in an *uniform* infrastructure
- Improve return of IT investment
- Cost minimization
- Performance/Usage maximization

Scheduling

- Centralized meta-scheduler
- Enforcement of Grid-wide policies (e.g. resource usage)

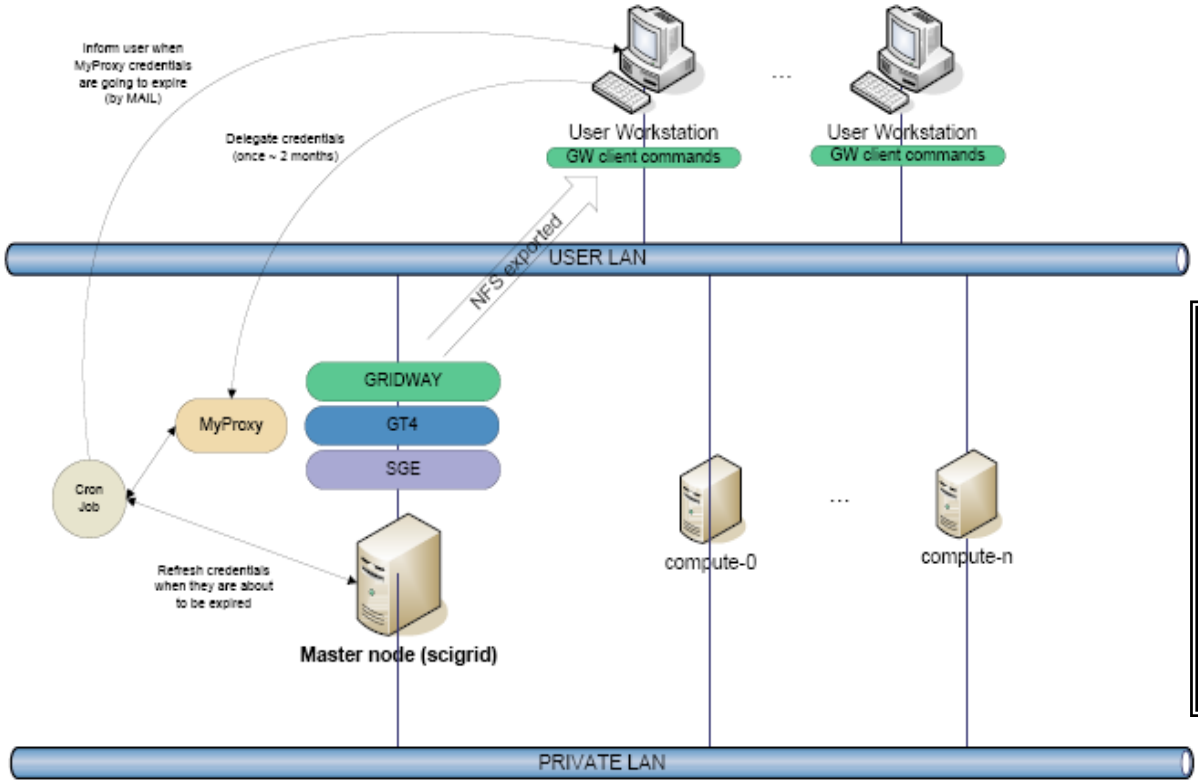
Deploying Single Meta-Scheduler Grids with GridWay



Single Meta-Scheduler Grids: Examples

European Space Astronomy Center

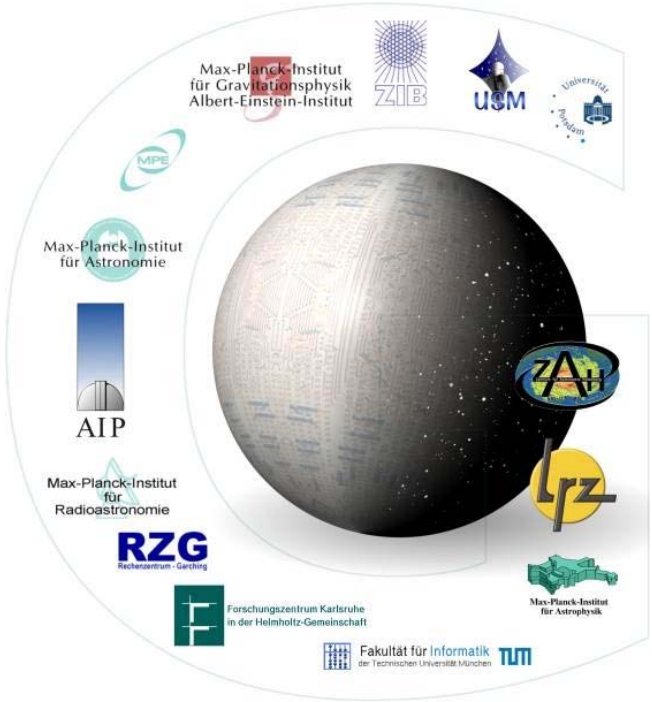
- Data Analysis from space missions (DRMAA)
- Site-level meta-scheduler
- One cluster 20 CPUs, 60 GB main memory

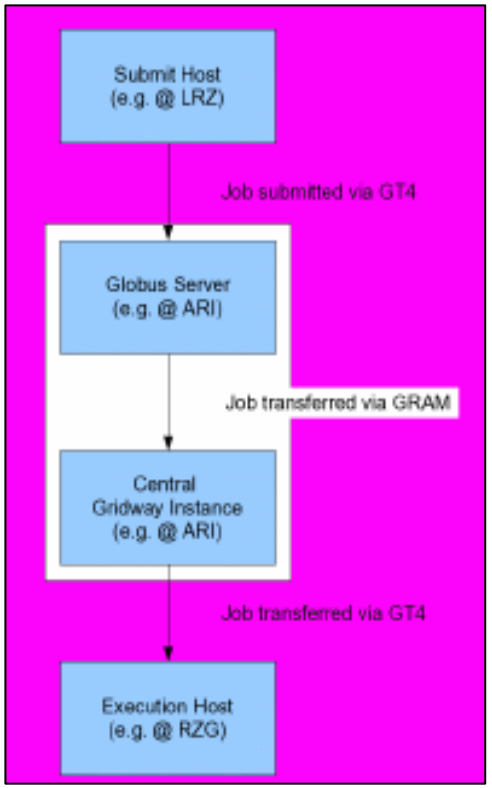


Single Meta-Scheduler Grids: Examples

AstroGrid-D, German Astronomy Community Grid

- Collaborative management of supercomputing resources & astronomy-specific resources
- Grid-level meta-scheduler (GRAM interface)
- 22 resources @ 5 sites, 800 CPUs

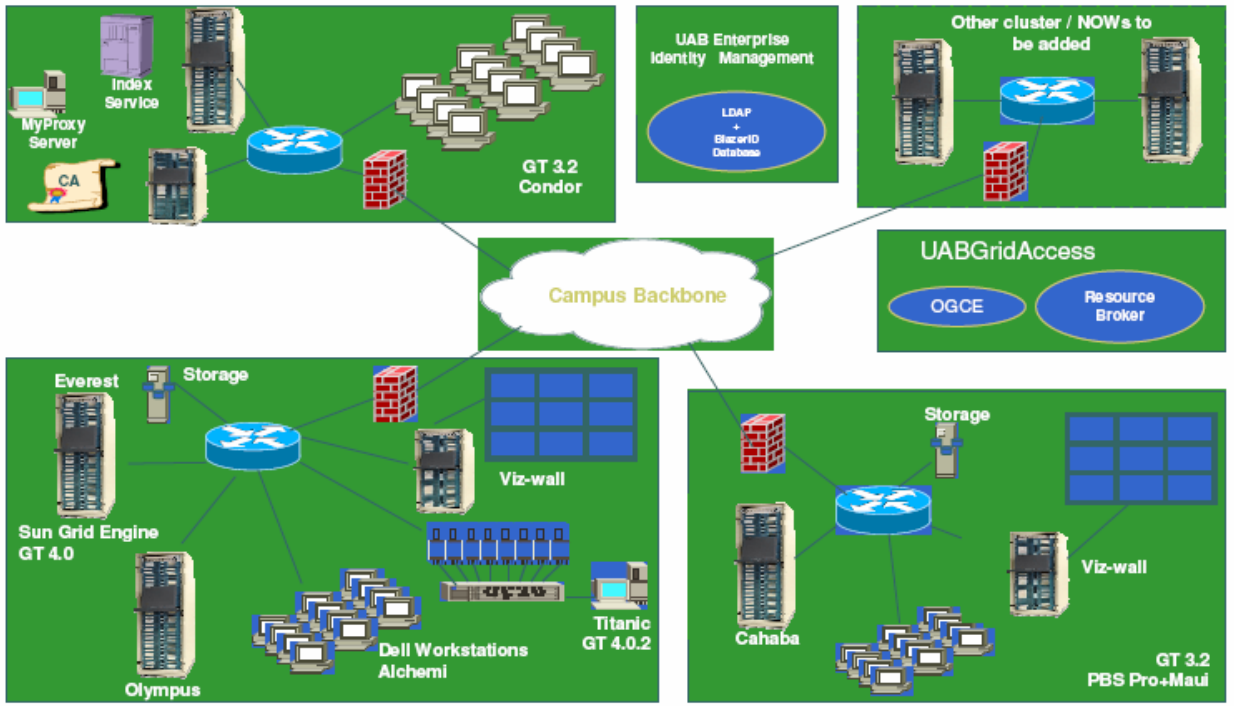




Single Meta-Scheduler Grids: Examples

UABGrid, University of Alabama at Birmingham

- Bioinformatics applications
- Campus-level meta-scheduler
- 3 resources (PBS, SGE and Condor)



Multiple Meta-Scheduler Grids

Characteristics

- Multiple meta-scheduler instances with access to multiple administration domains (different organizations or partners)
- Large scale, loosely-coupled infrastructures
- Shared by several Virtual Organizations (VO)

Goal & Benefits

- Large-scale, secure and reliable sharing of resources
- Support collaborative projects
- Access to higher computing power to satisfy peak demands

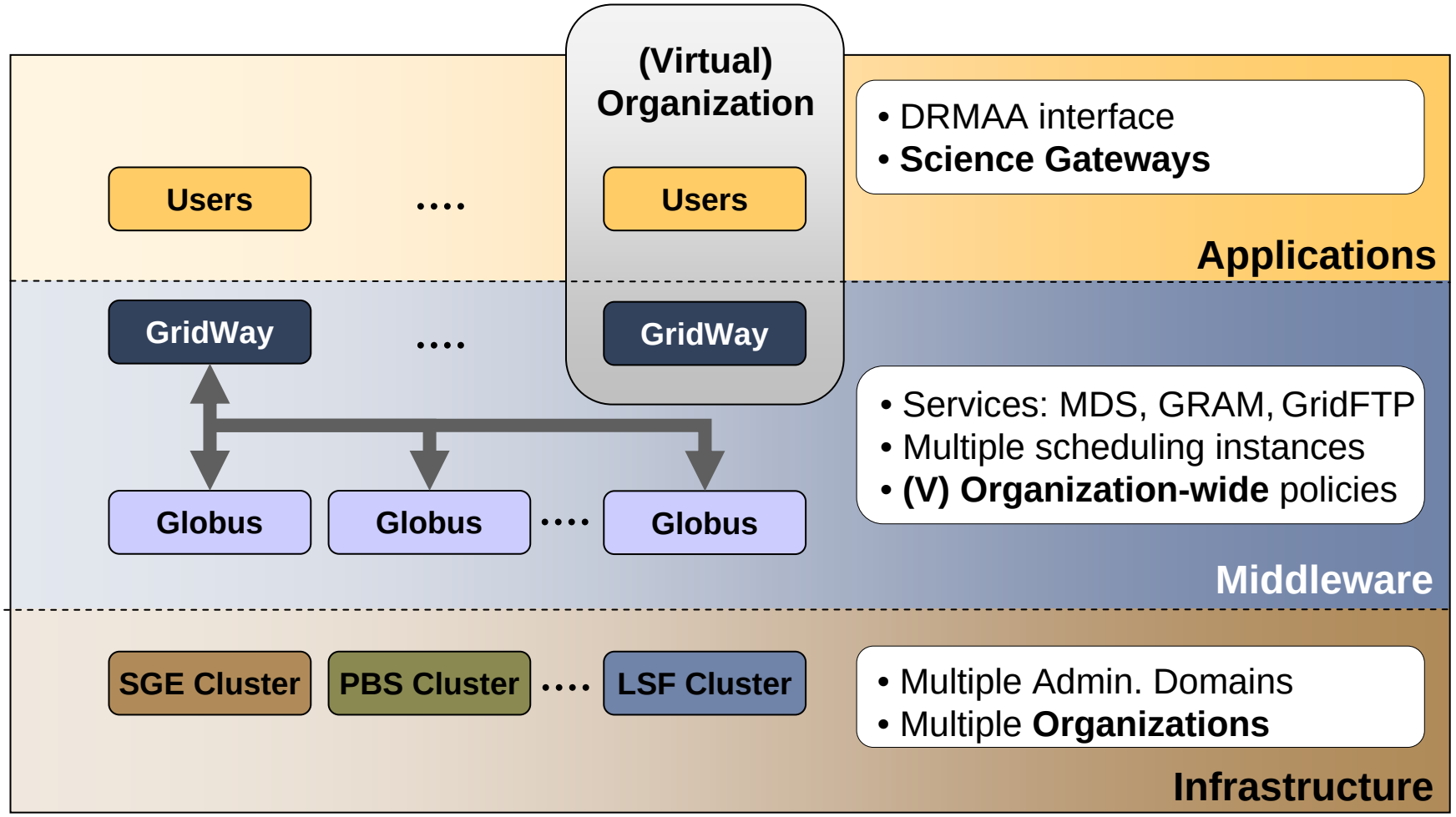
Scheduling

- Decentralized scheduling system
- Enforcement of organization-wide policies

3. A Taxonomy for Grid Computing Infrastructures

3.1. Multiple Administration Domains

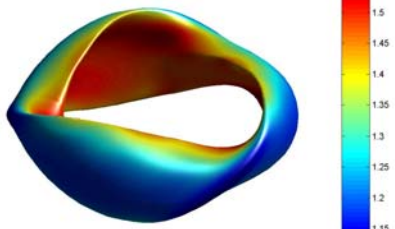
Deploying Single Meta-Scheduler Grids with GridWay



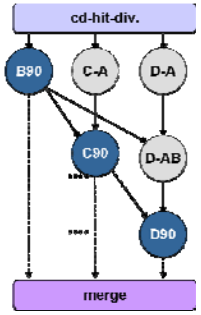
3. A Taxonomy for Grid Computing Infrastructures

3.1. Multiple Administration Domains

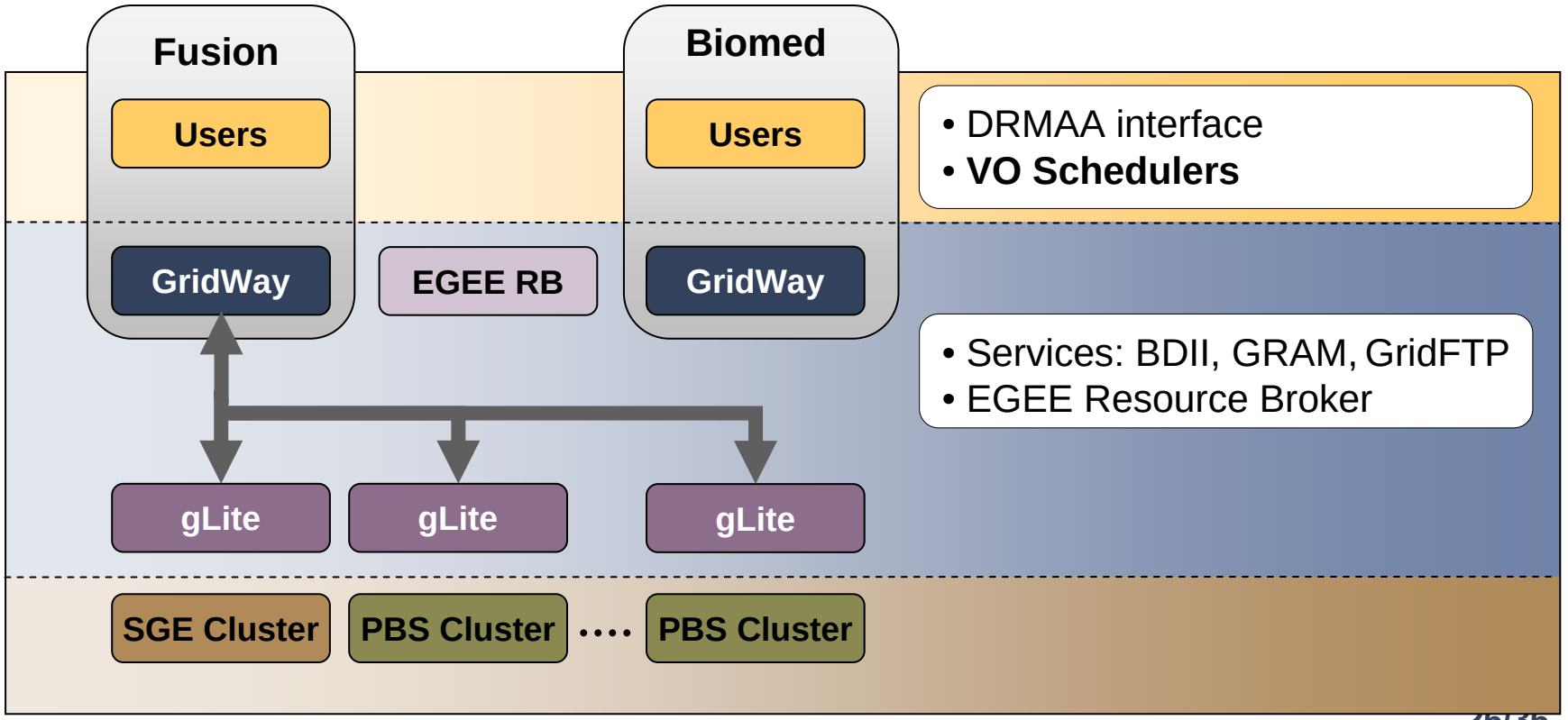
Multiple Meta-Scheduler Grids: Examples



Massive **Ra**y **Tra**cing



CD-HIT workflow



Single Meta-Scheduler Layer Grids

Characteristics

- Single layer (one ore more meta-schedulers) with *plain* access to the underlying Grids
- Access multiple Grid admin. domains
- Based on different middleware stacks

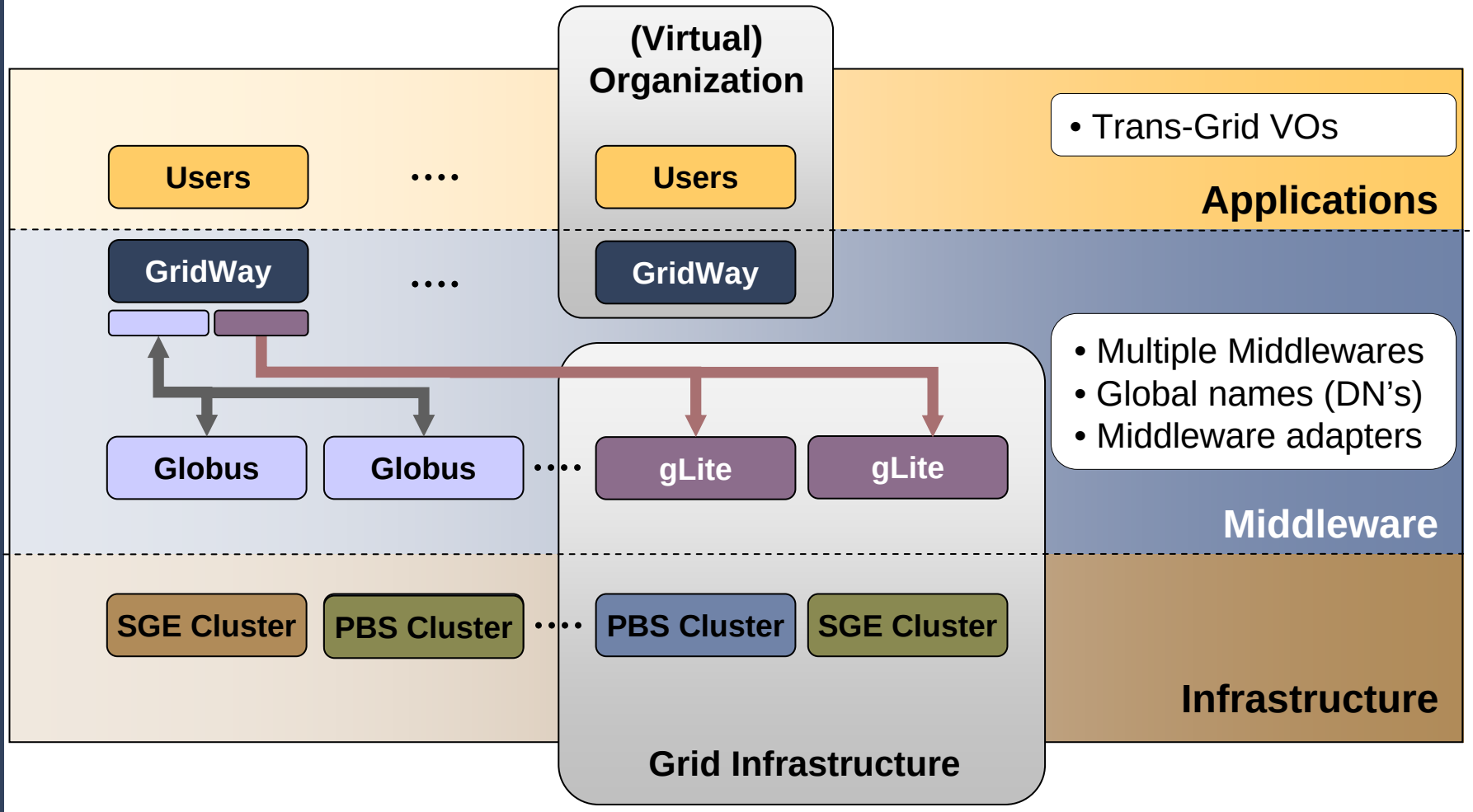
Goal & Benefits

- Integrate multiple Grids in an single infrastructure
- Collaboration between trans-grid VOs

Scheduling

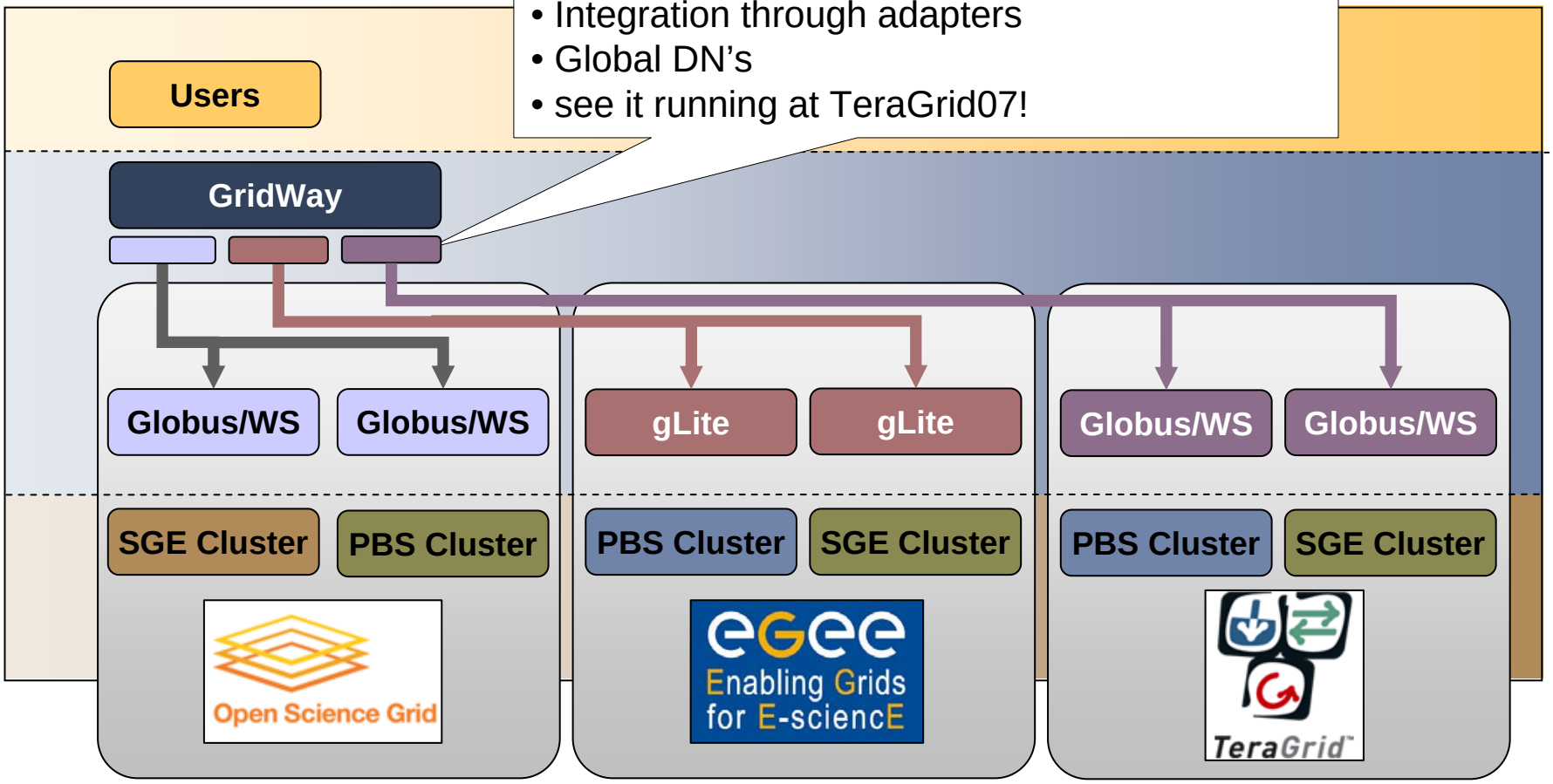
- Enforcement of organization-wide Grid-aware policies
- Adapters to interface different middleware stacks

Deploying Single Meta-Scheduler Layer Grids with GridWay



Single Meta-Scheduler Layer Grids: Example

- Different Middlewares (e.g. WS and pre-WS)
- Different Data/Execution architectures
- Different Information models
- Integration through adapters
- Global DN's
- see it running at TeraGrid07!



Multiple Meta-Scheduler Layer Grids

Characteristics

- Multiple meta-scheduler layers in a hierarchical structure
- Use standard interfaces to virtualize a Grid infrastructure
- Resource provision in a utility fashion (provider/consumer)

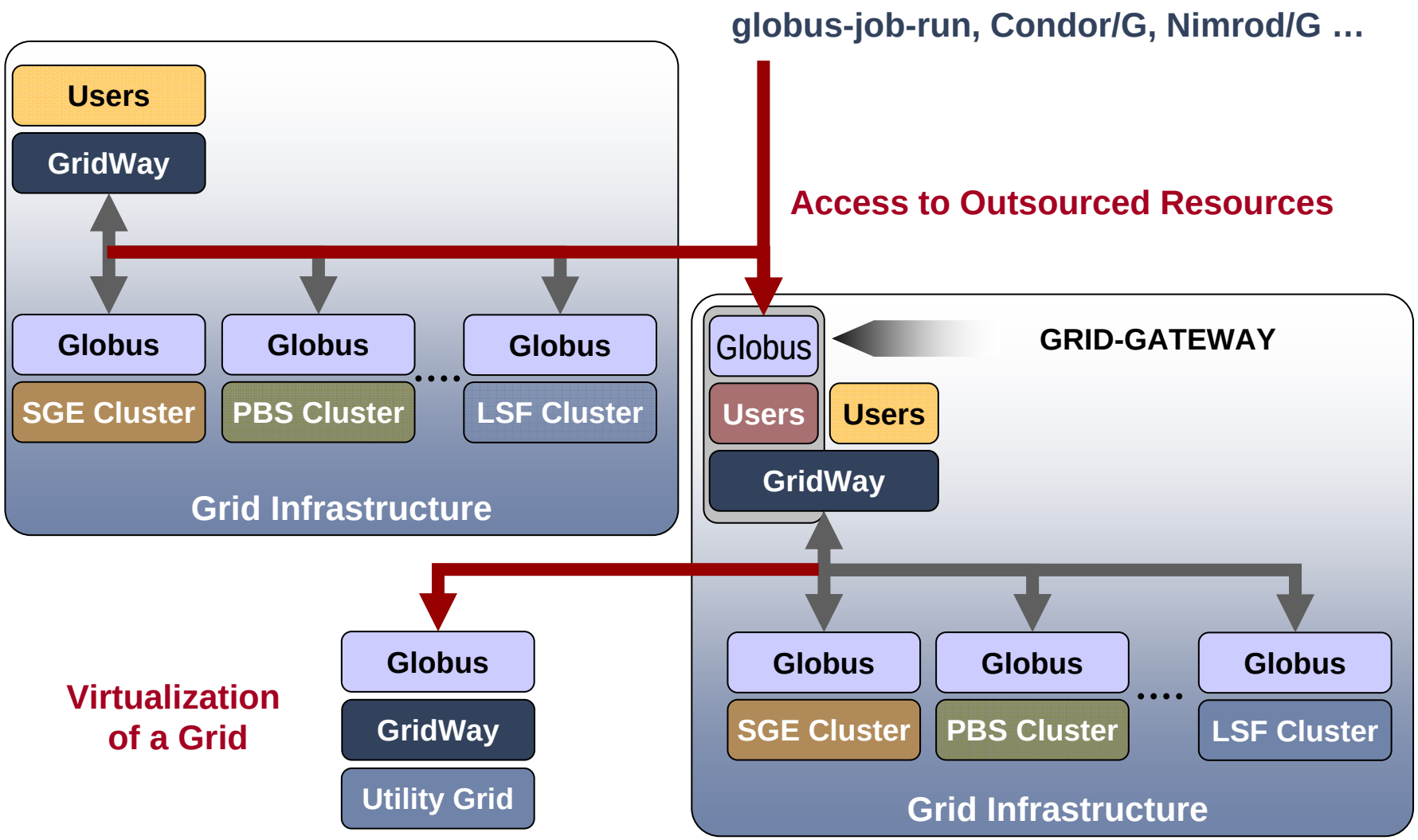
Goal & Benefits

- Supply resources on-demand, making resource provision more agile and adaptive.
- Access to *unlimited* computational capacity
- Transform IT costs from fixed to variable
- Seamlessly integration of different Grids (The Grid)

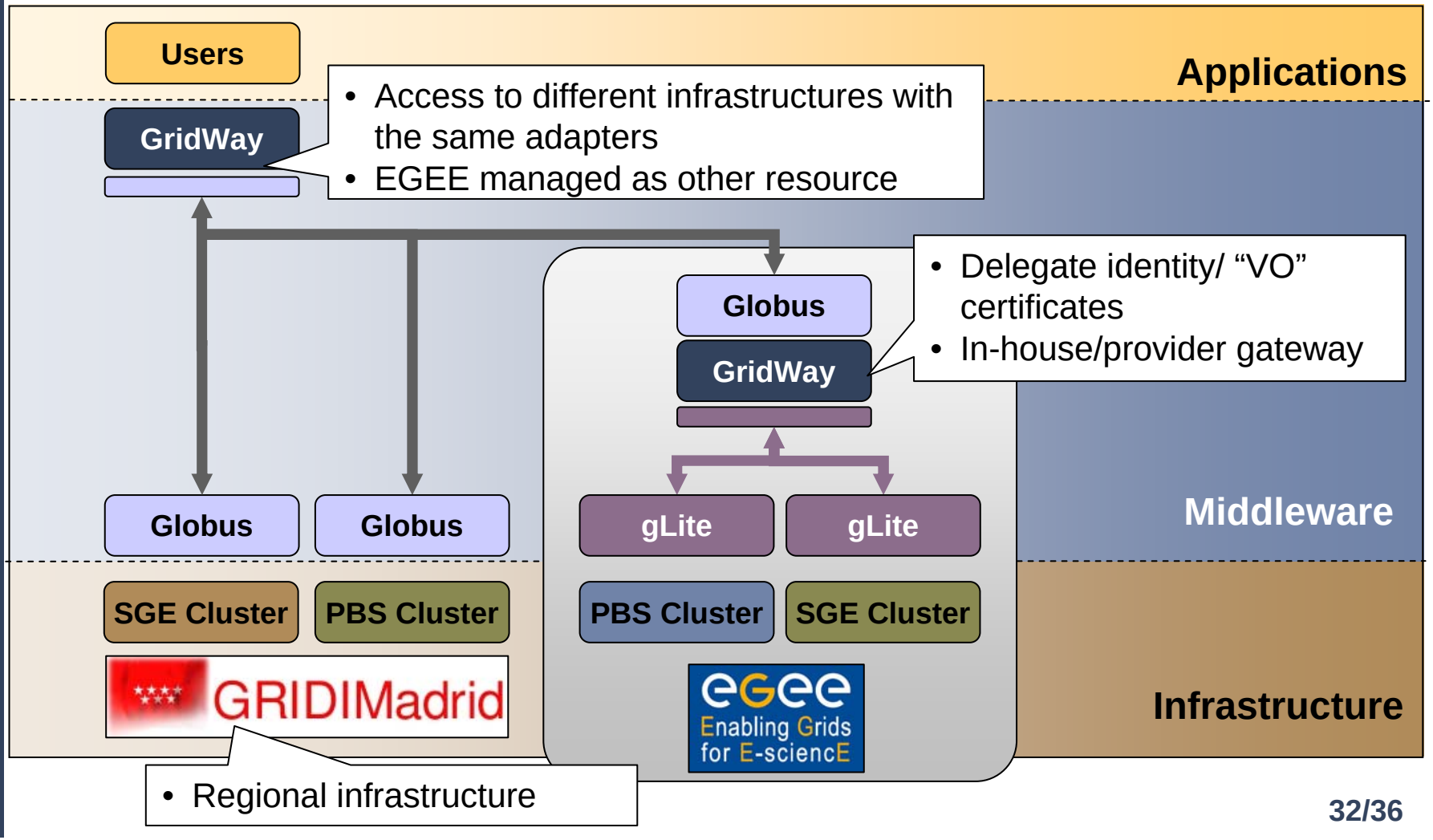
Scheduling

- Each Grid is handled as any other resource
- Characterization of a Grid as a single resource

Deploying Multiple Meta-Scheduler Layer Grids with GridWay



Multiple Meta-Scheduler Layer Grids: Example



Contents

1. Computing Resources
2. Grid Middleware
3. A Taxonomy for Grid Computing Infrastructures
4. **A Note on Usability**

4. A Note on Usability

Use of Grid Infrastructures

Grid specific commands & API's

- Applications must be ported to the Grid
- Process (submission, monitoring…) must be adapted to the Grid
- New interfaces (e.g. portal) to simplify Grid use

LRMS-like commands & API's

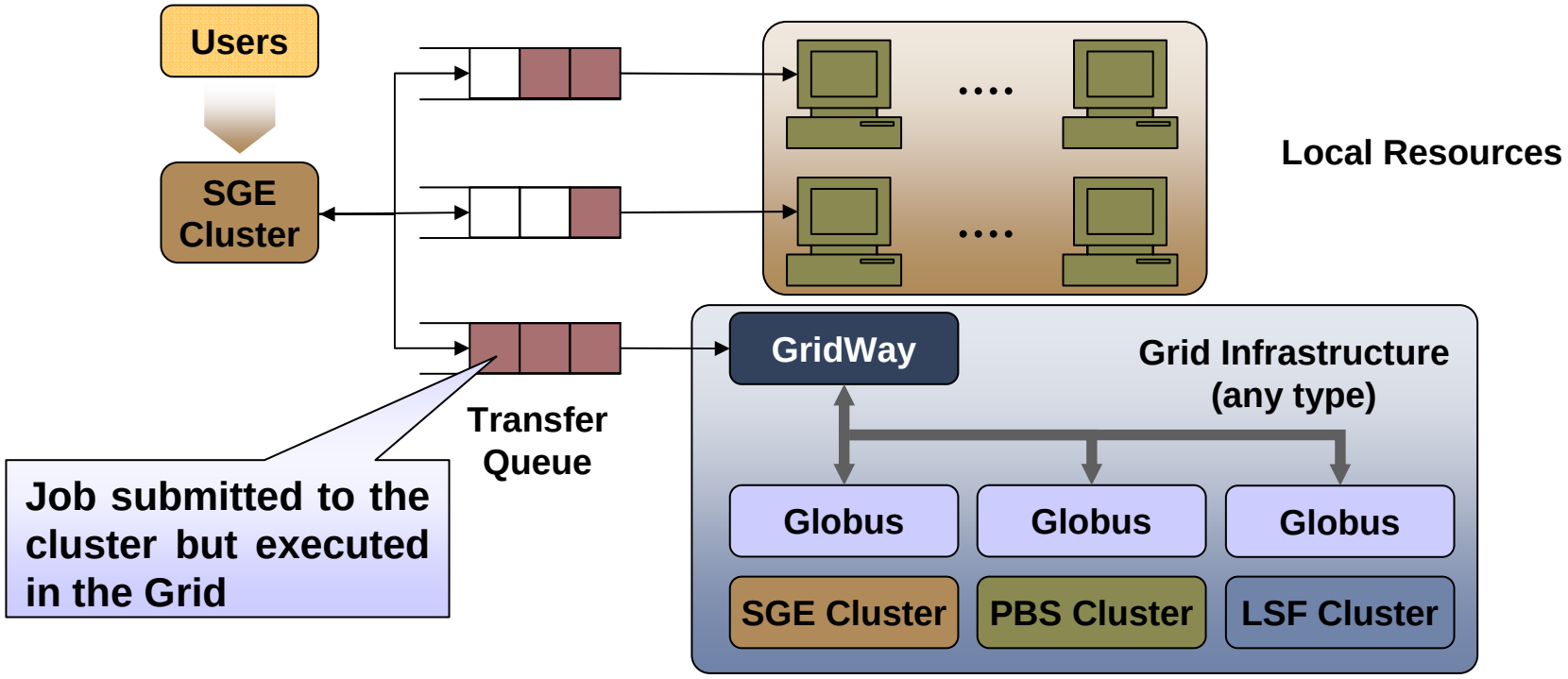
- A familiar environment to interact with a computational platform
- Some systems provide LRMS-like environment for Computational Grids (e.g. GridWay)
- Process still need to be adapted
- Applications would greatly benefit from standards (DRMAA)

Transfer Queues: Seamless integration of a Grid

4. A Note on Usability

Transfer Queues: Seamless integration of a Grid

- Communicate LRM systems with meta-schedulers (the other way)
- Users keep using the same interface, even applications (e.g. DRMAA)



Thank you for your attention!