the globus alliance

GridWay

Grid Scheduling Architectures with Globus

Workshop on Scheduling WS 07
Cetraro, Italy
July 28, 2007

Ignacio Martin Llorente
Distributed Systems Architecture Group
Universidad Complutense de Madrid

Universidad Complutense Madrid

La Suma de Todos
Consejería de Educación
Comunidad de Madrid
www.madrid.org

MINISTERIO DE EDUCACION Y CIENCIA

Contents

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 ← **Decentralized Decoupled**

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

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

High Performance Computing (SMP, MPP, Clusters)

High Throughput Computing (SMP, MPP, Clusters, Network Systems)

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*

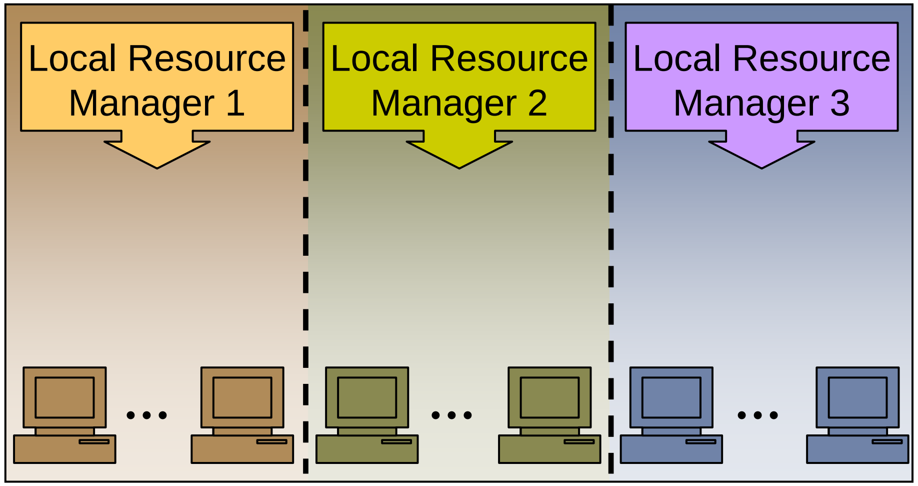
Independent Suppliers	*Open Source*	*OEM Proprietary*
2 *Platform Computing* 3 LSF	2 *Altair* Open PBS	1 *IBM* Load Leveler
2 *Altair* PBS Pro	3 *University of Wisconsin* Condor	1 *Cray* NQE
	2 *Sun Microsystems* 3 SGE	

1. Computing Resources

1.3. Local Resource Management Systems

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

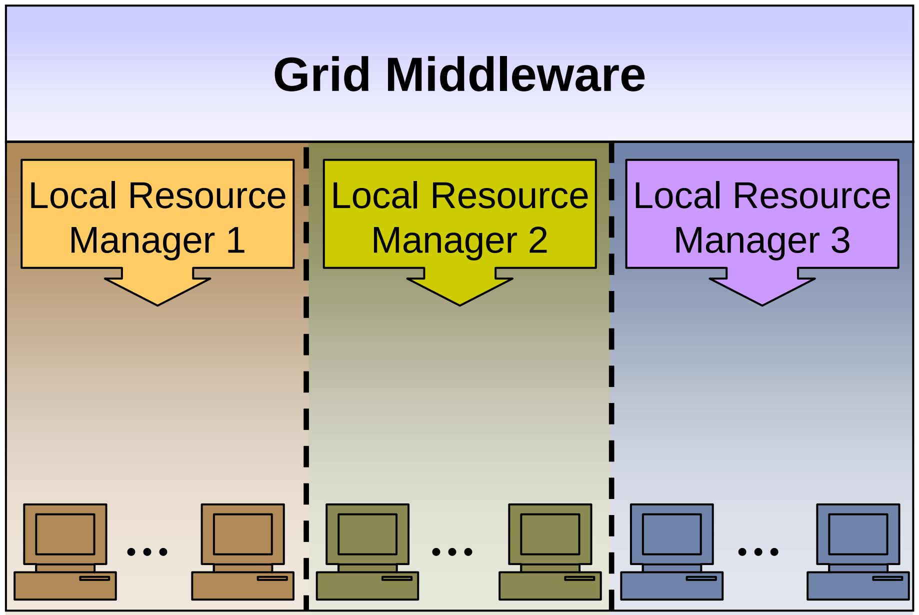
2. Grid Middleware

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 for Each Type of Resources: User can access a wide set of resources.

Types of Resources: Computational, storage and network.

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 Software Distributions**

gLite — glite.web.cern.ch

UNICORE — www.unicore.org

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

GRIA — www.gria.org

VDT Virtual Data Toolkit — vdt.cs.wisc.edu

- **Open Source Software Communities**

the globus alliance — **The Globus Alliance** (dev.globus.org)

The Globus Alliance Community

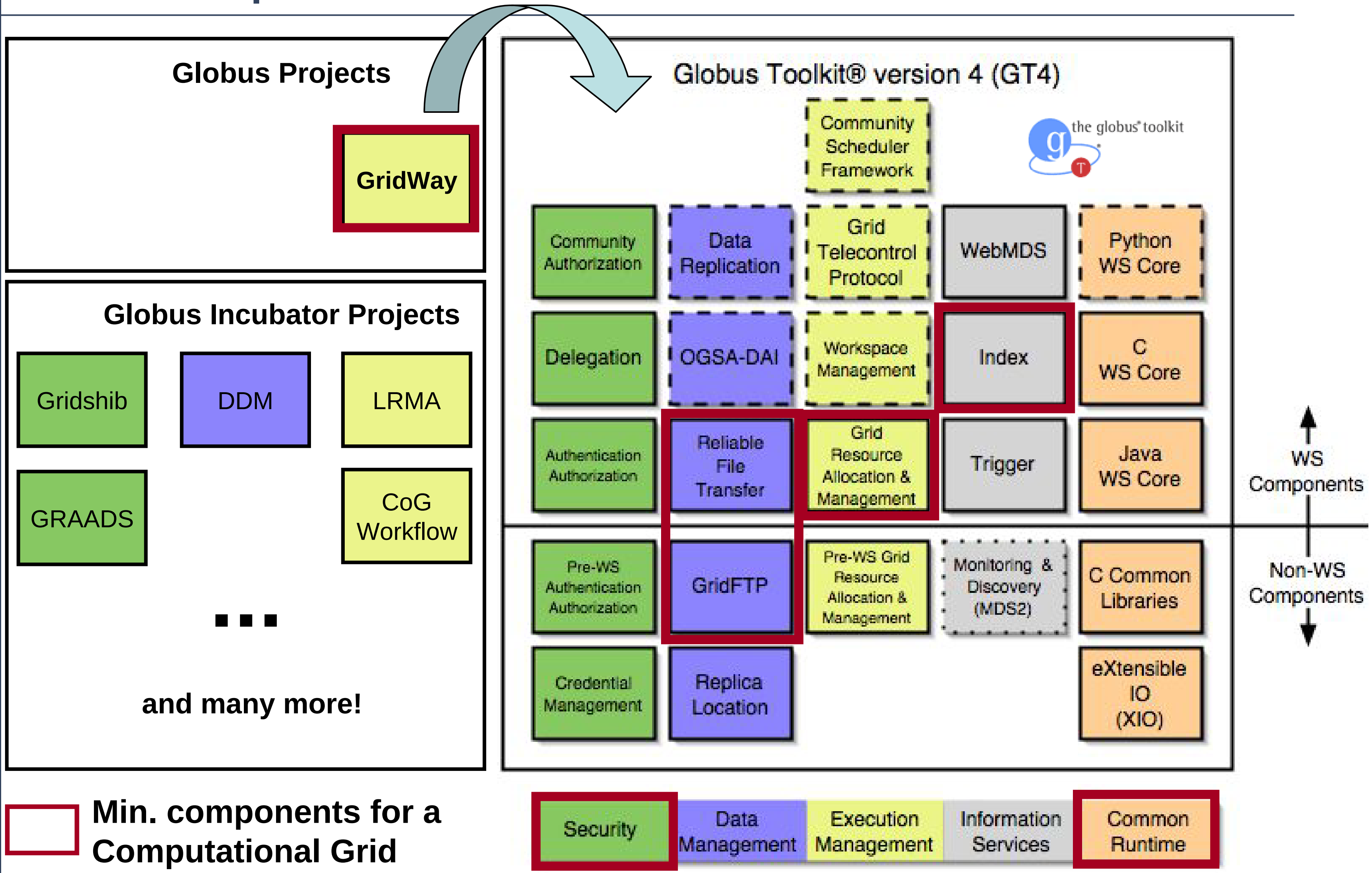
Open-Source Software Community =
Open-Source Software + Open Development Processes

- **Open Community Project** based on Apache Jakarta model:
  - Control of each individual project is in hands 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

- Software distribution that integrates a selected group of Globus technologies
- GT **provides basic services** to allow secure remote operation over multiple administrative domains with different LRM systems and access policies.

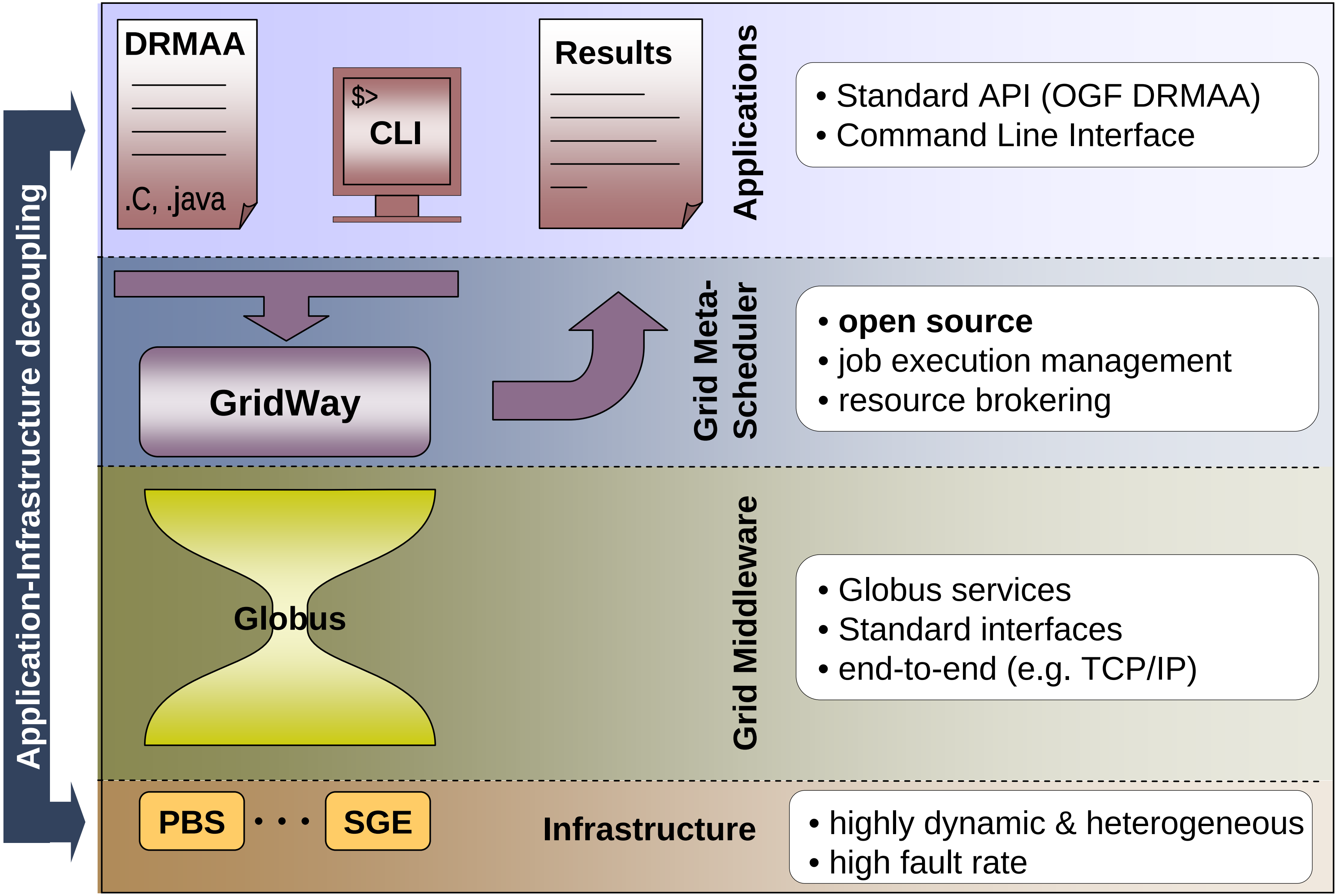
Globus Components



2. Grid Middleware

2.3. The GridWay Meta-scheduler

Global Architecture of a Computational Grid



GridWay

DSA Group

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

2.3. The GridWay Meta-scheduler

Features

Workload Management

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

User Interface

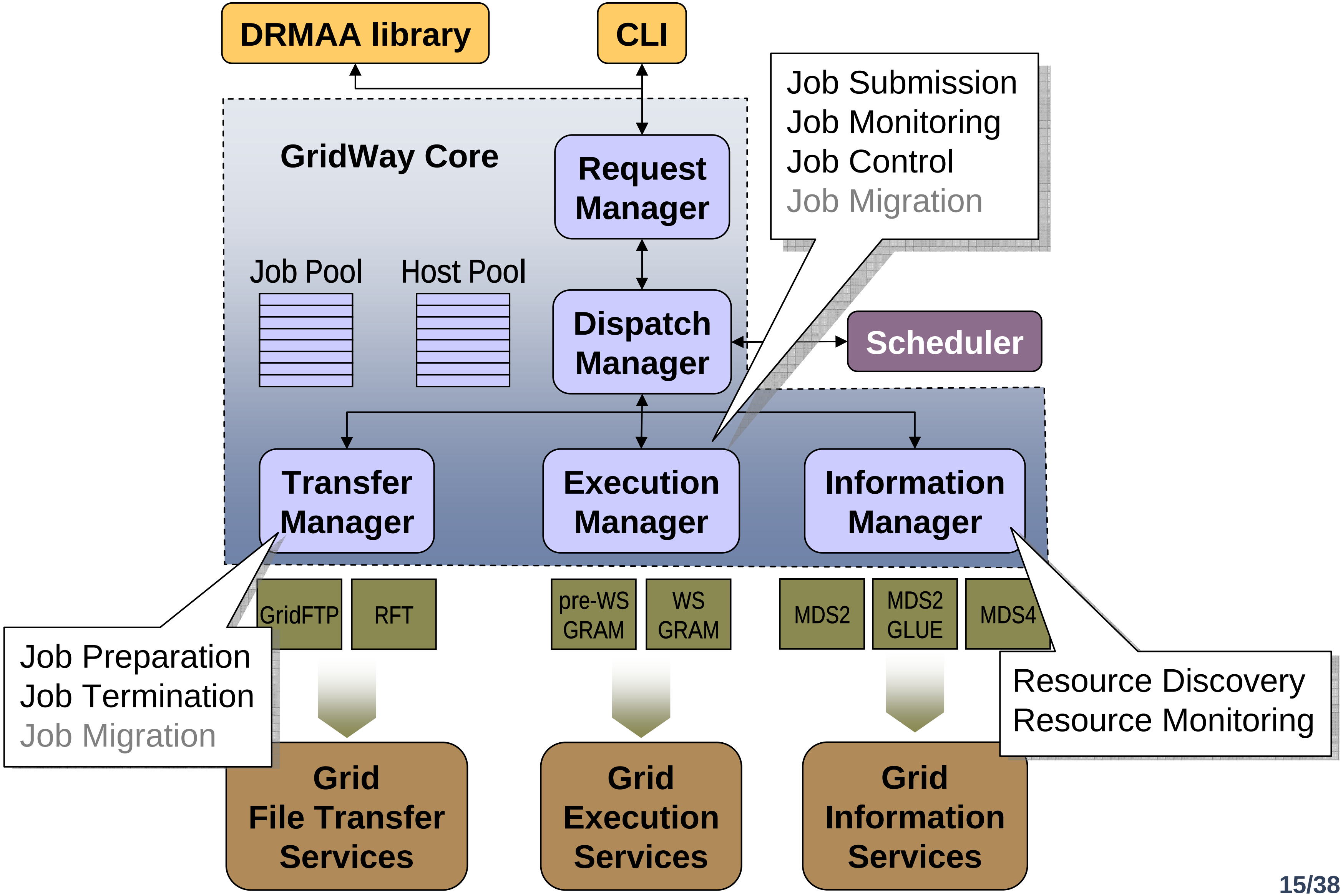
- OGF standards: JSDL & DRMAA (C and JAVA)
- Analysis of trends in resource usage
- Command line interface, similar to that found on local LRM Systems

Integration

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

2.3. The GridWay Meta-scheduler

GridWay Internals



GridWay

2. Grid Middleware

2.3. The GridWay Meta-scheduler

Grid-specific Scheduling Policies

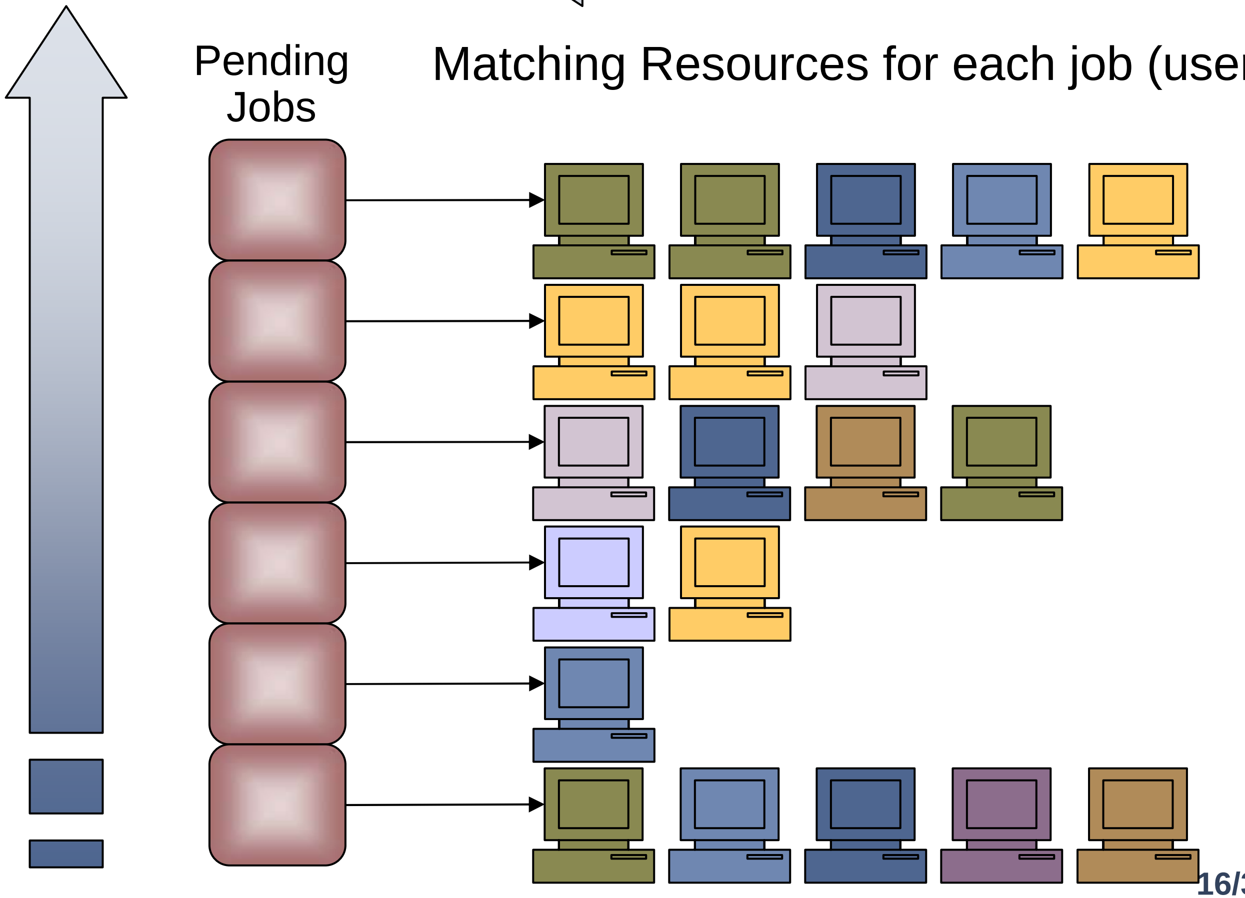
Resource Policies

- Rank Expressions
- Fixed Priority
- User Usage History
- Failure Rate

Grid Scheduling = Job + Resource Policies

Job Policies

- Fixed Priority
- Urgent Jobs
- User Share
- Deadline
- Waiting Time



DSA Group

2. Grid Middleware

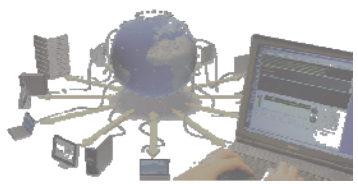
2.3. The GridWay Meta-scheduler

Centralized Coupled

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

Decentralized Decoupled

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

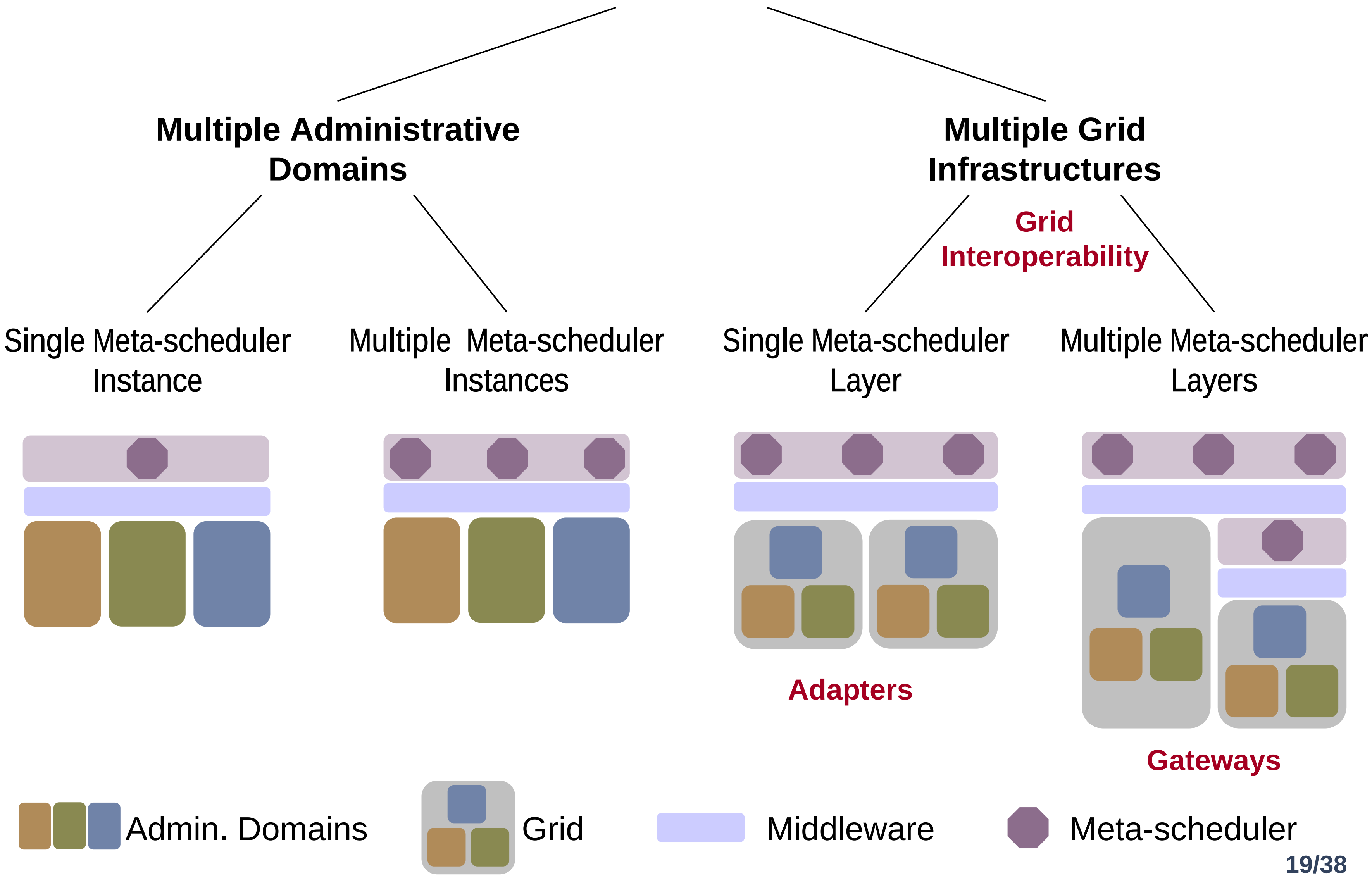
High Performance Computing

High Throughput Computing

Contents

3. A Taxonomy for Grid Architectures

3.1. The Taxonomy



Single Meta-Scheduler Grids

Characteristics

- One meta-scheduler instance with access to resources that may belong to different administrative domains
- Small scale infrastructures (campus or enterprise) that may be geographically distributed in different sites

Goal & Benefits

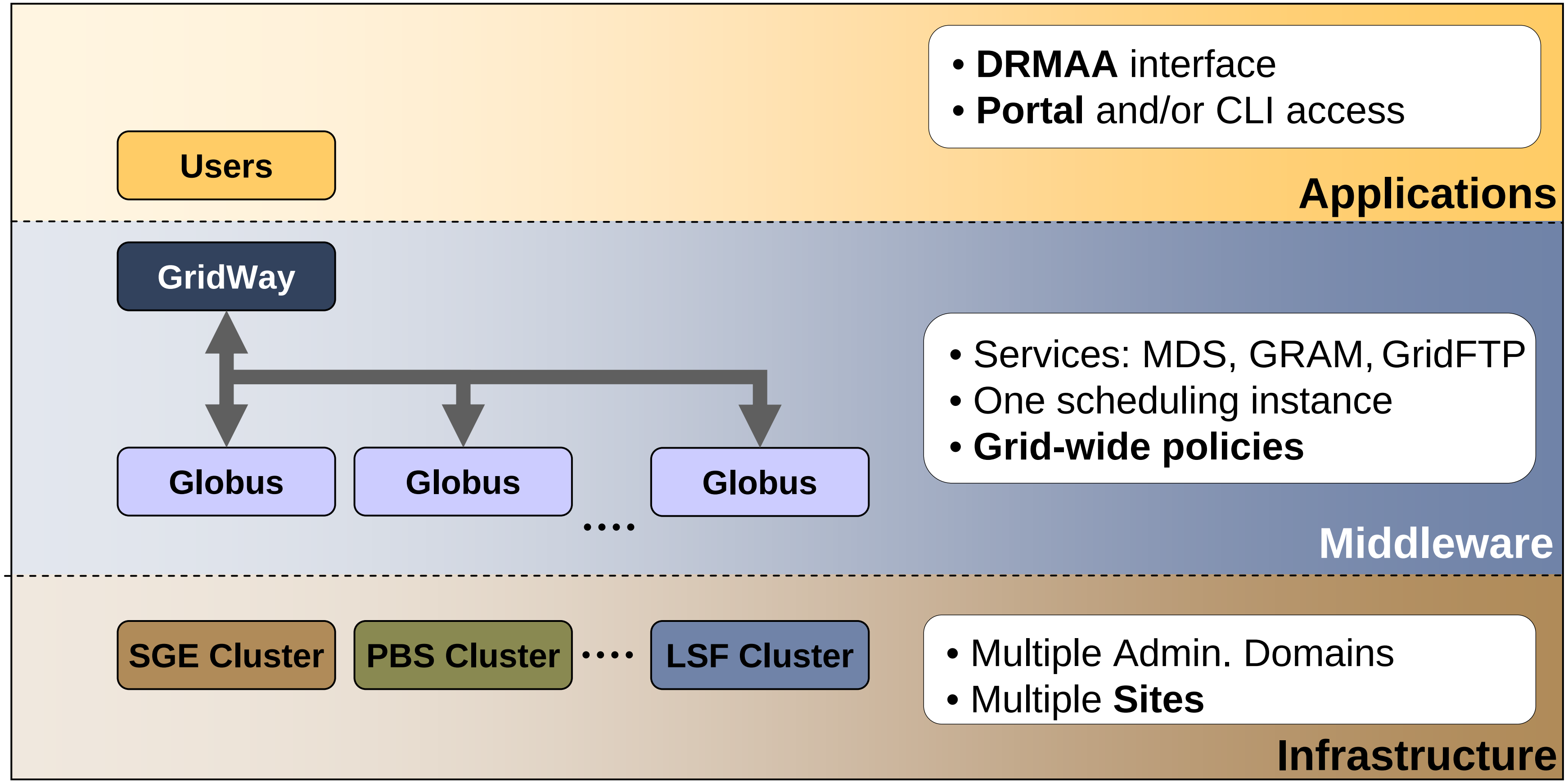
- Integrate multiple heterogeneous systems and/or administrative domains in an *uniform/centralized* infrastructure
- Improve return of IT investment
- Cost minimization
- Performance/Usage maximization

Scheduling

- Centralized meta-scheduler that allows the enforcement of **Grid-wide policies** (e.g. resource usage)

3.2. Multiple Administrative Domains

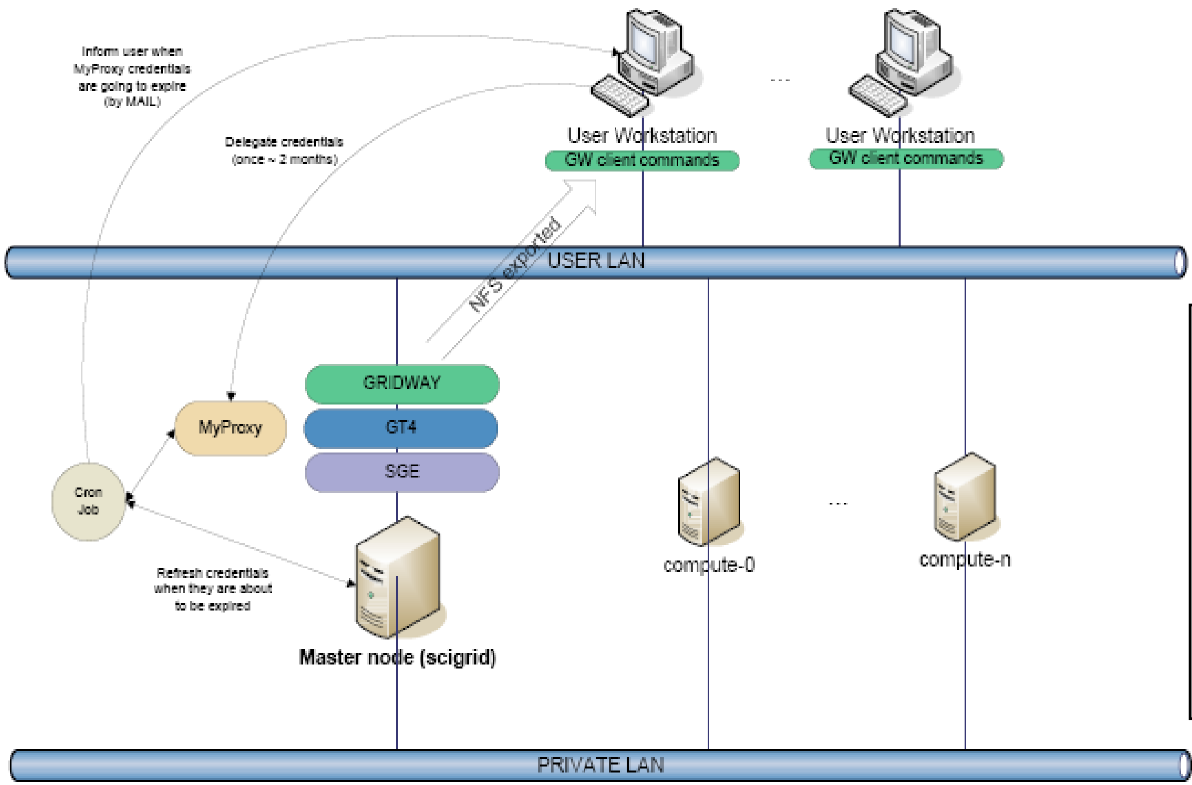
Deploying Single Meta-Scheduler Grids with GridWay



Single Meta-Scheduler Grids: Examples

European Space Astronomy Center

- Data Analysis from space missions (DRMAA)
- Site-level meta-scheduler
- Several clusters



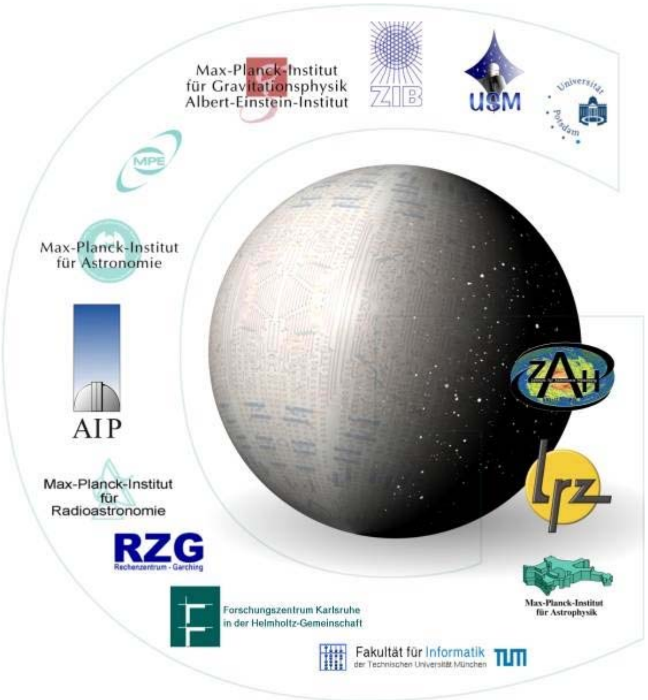
3. A Taxonomy for Grid Scheduling Architectures

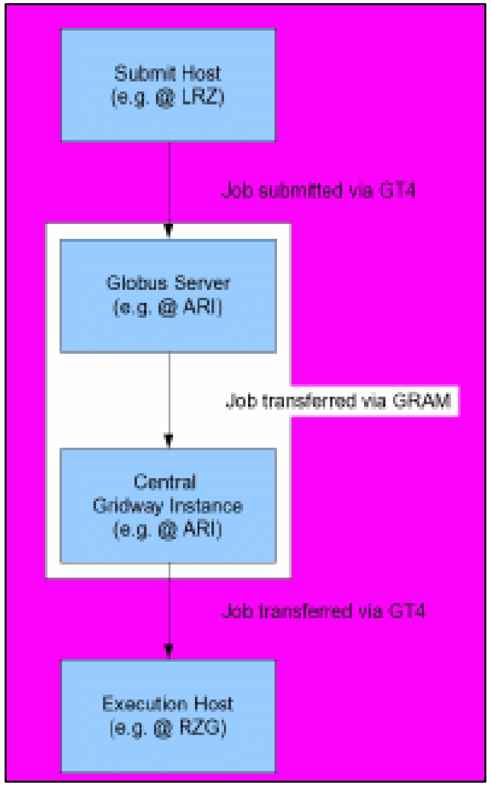
3.2. Multiple Administrative Domains

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



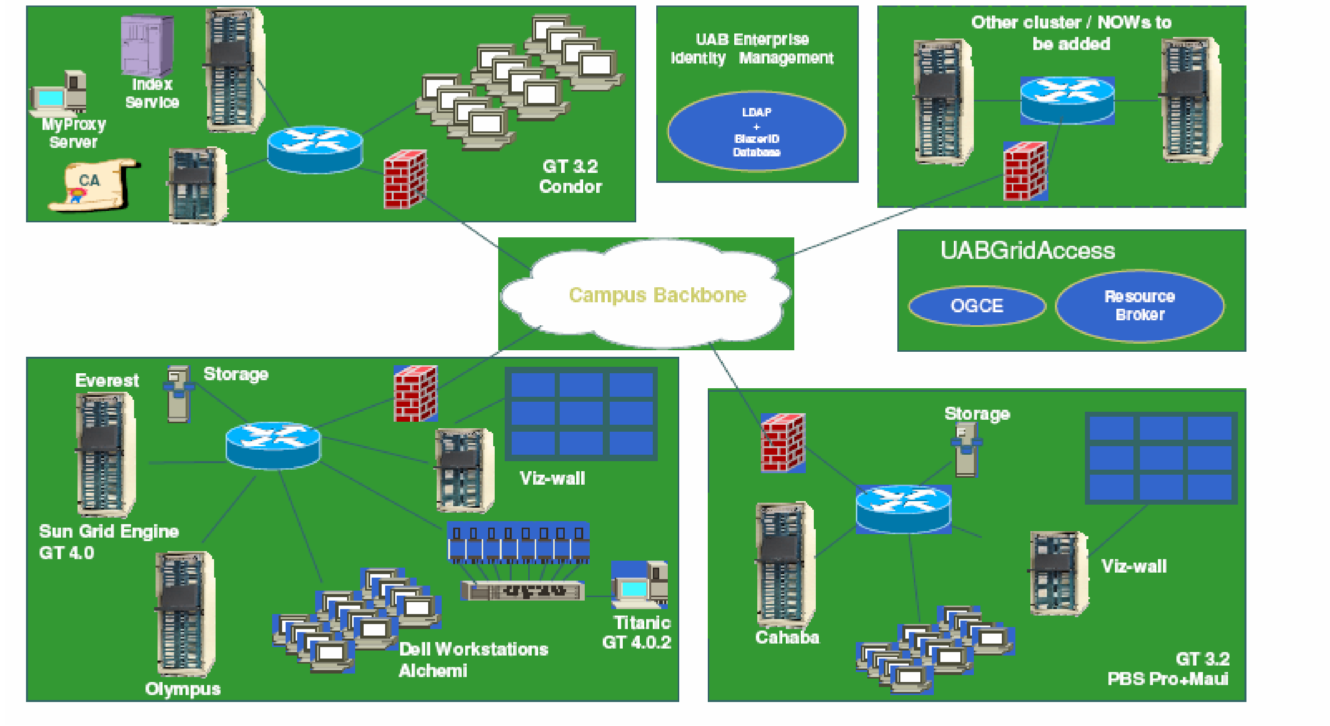


Single Meta-Scheduler Grids: Examples

UABGrid, University of Alabama at Birmingham

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

THE UNIVERSITY OF ALABAMA AT BIRMINGHAM



Multiple Meta-Scheduler Grids

Characteristics

- Multiple meta-scheduler instances with access to resources belonging to different administrative domains (different organizations or partners)
- Large scale, loosely-coupled infrastructures (Partner Grids) shared by several Virtual Organizations

Goal & Benefits

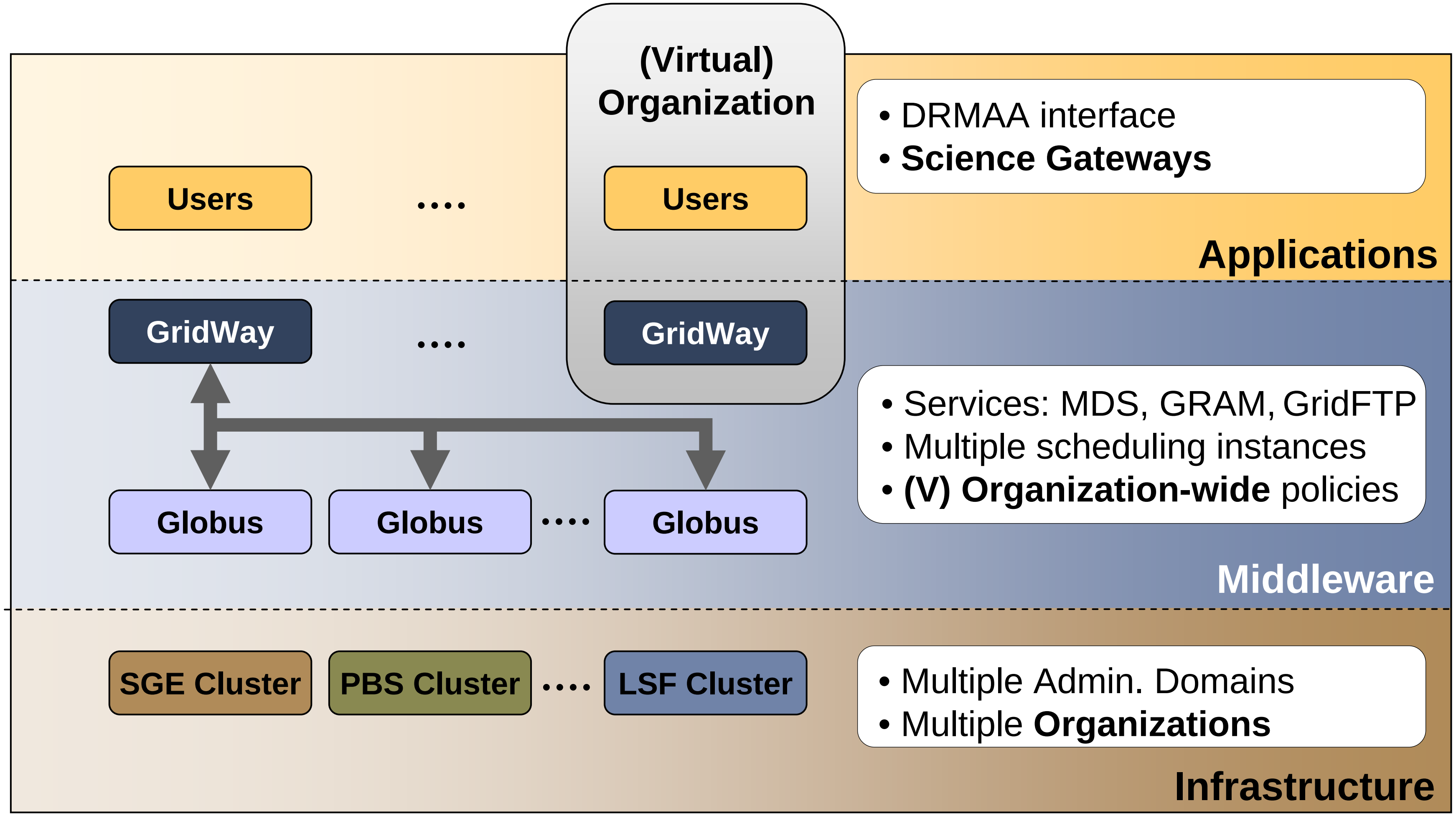
- Large-scale, secure and reliable sharing of resources
- Support collaborative projects
- Access to higher computing power to satisfy peak demands

Scheduling

- Decentralized scheduling system that allows the enforcement of **organization-wide** policies

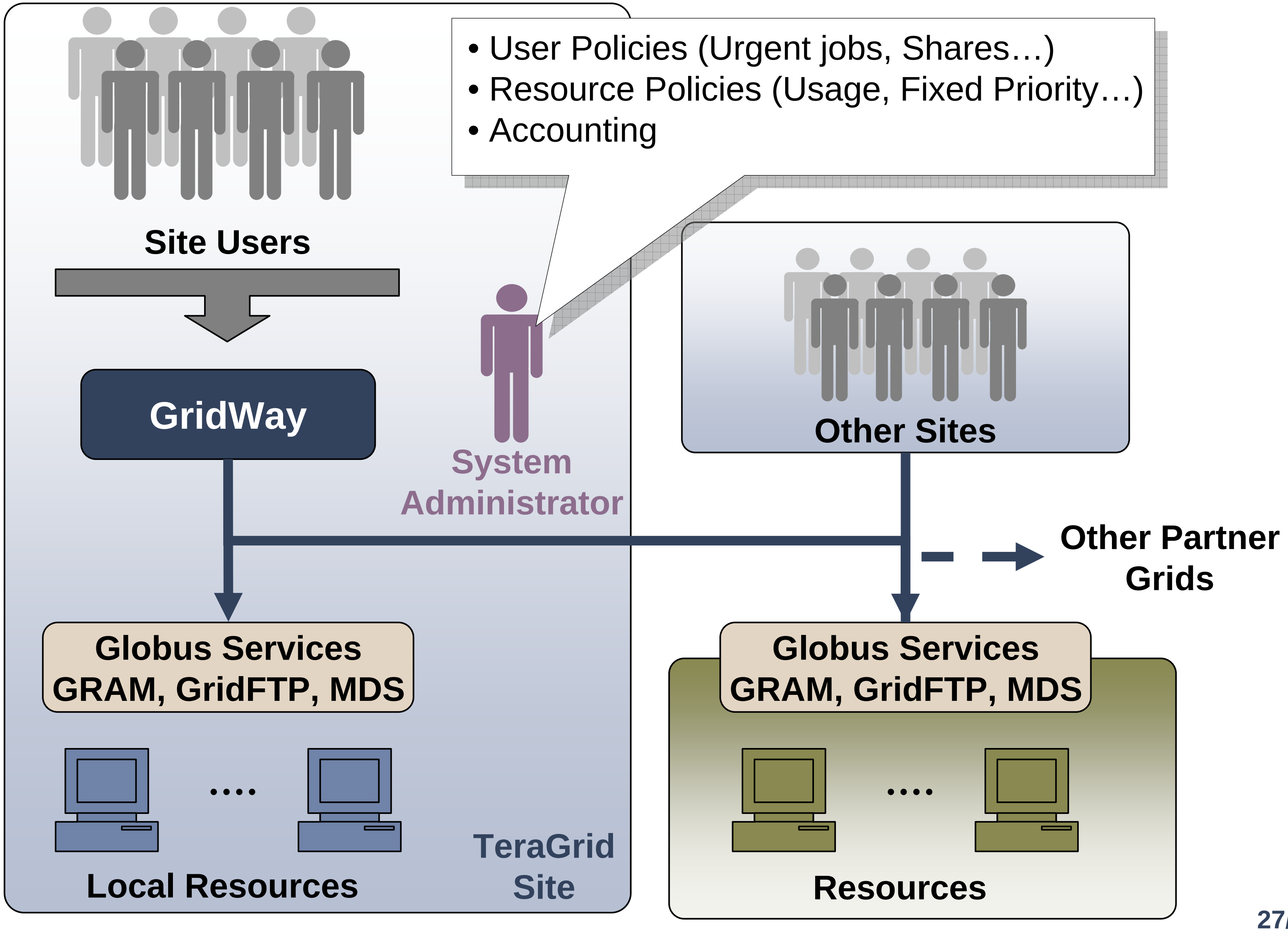
3.2. Multiple Administrative Domains

Deploying Multiple Meta-Scheduler Grids with GridWay



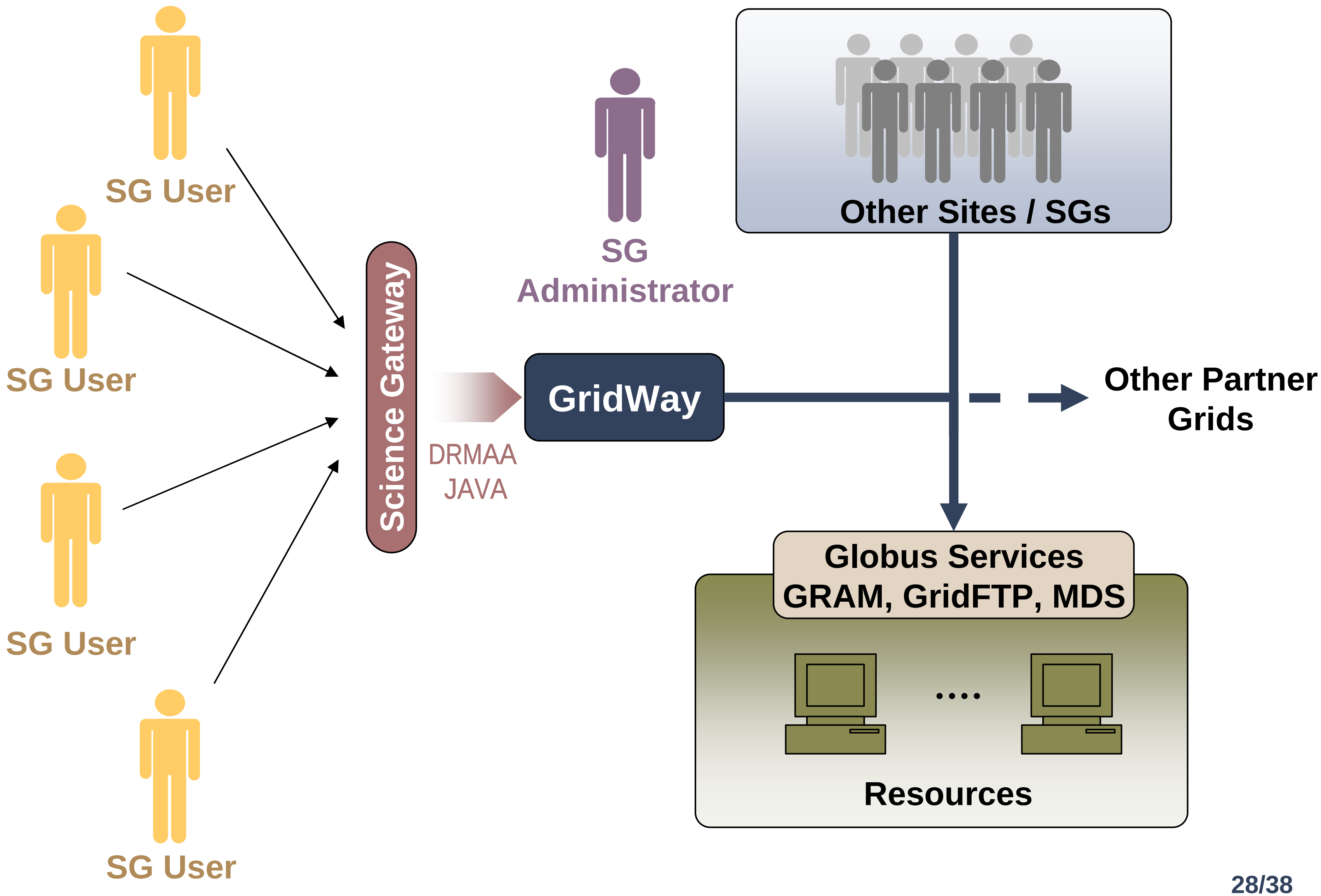
3.2. Multiple Administrative Domains

Multiple Meta-Scheduler Grids: Generic Examples



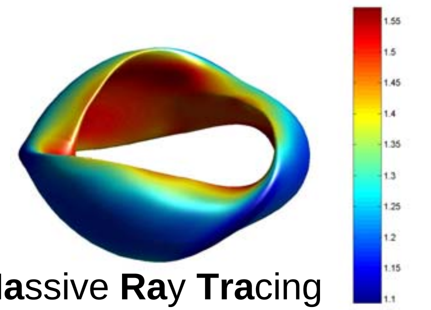
3.2. Multiple Administrative Domains

Multiple Meta-Scheduler Grids: Generic Examples

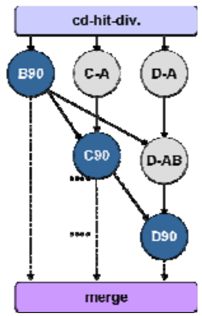


3.2. Multiple Administrative Domains

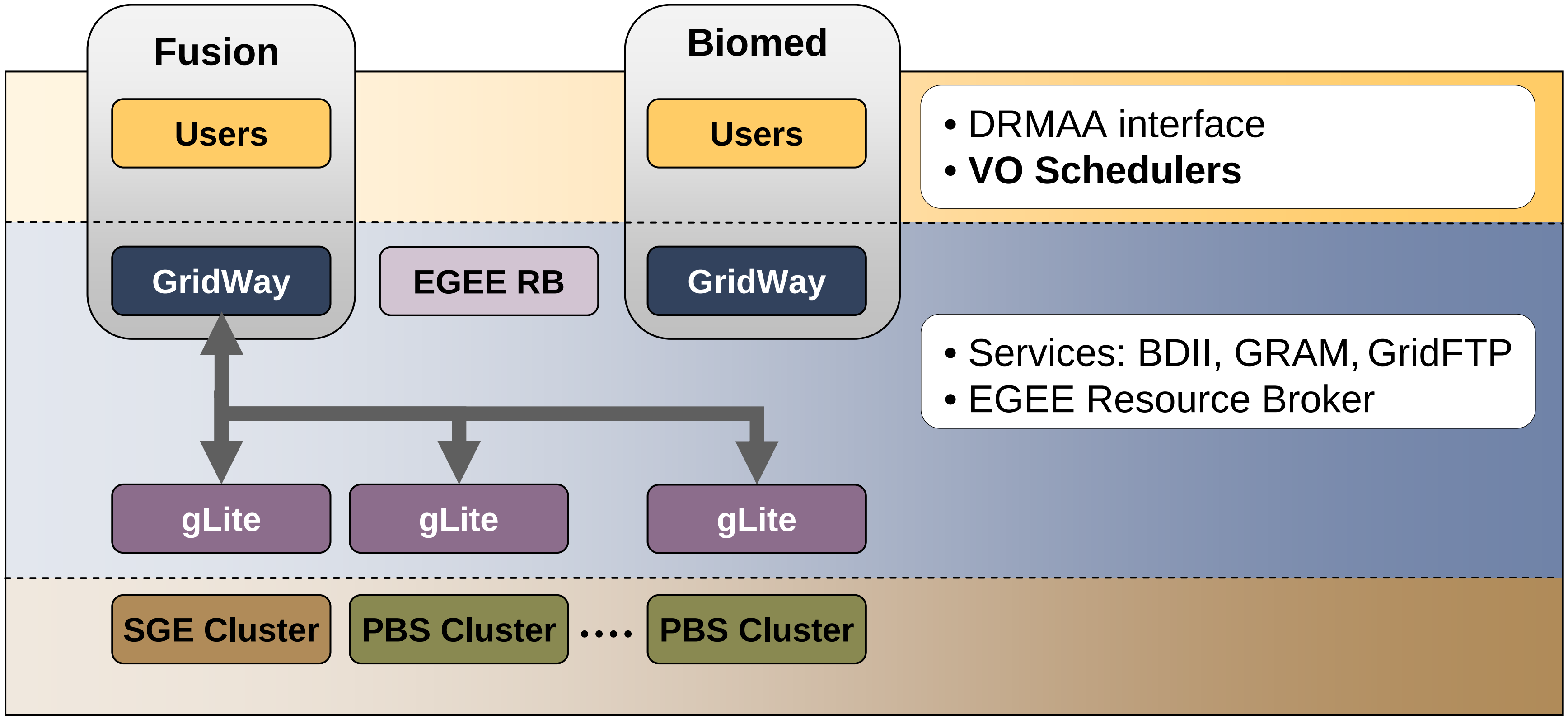
Multiple Meta-Scheduler Grids: Examples



Massive **Ray Tracing**



CD-HIT workflow



Single Meta-Scheduler Layer Grids

Characteristics

- Single layer (one ore more meta-schedulers) with *plain* access to the underlying Grids
- (Virtual) Organizations involved in different Grid infrastructures

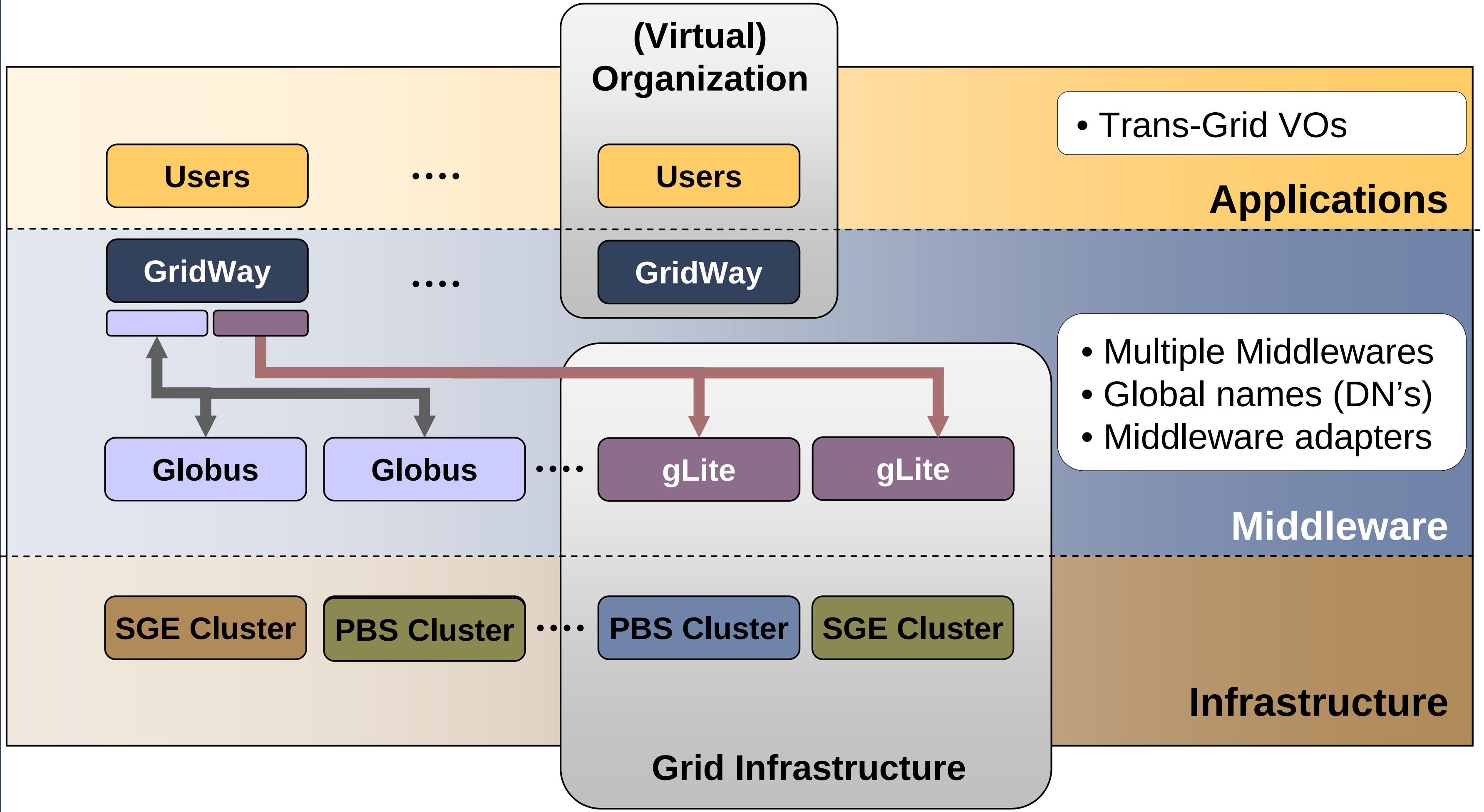
Goal & Benefits

- Integrate multiple Grids based on different middleware stacks
- Collaboration between trans-grid VOs

Scheduling

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

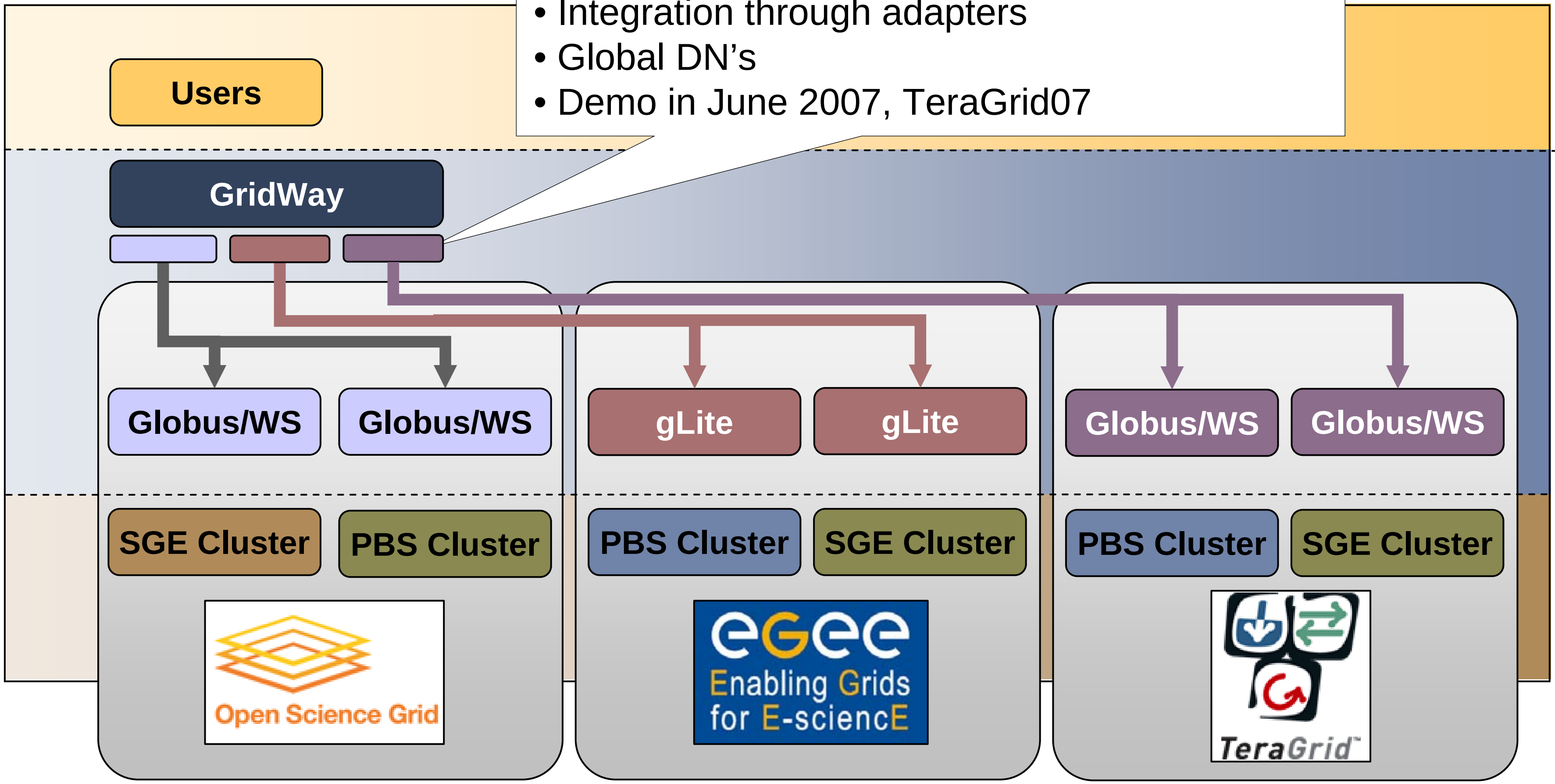
Deploying Single Meta-Scheduler Layer Grids with GridWay



3.3. Multiple Grid Infrastructures

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
- Demo in June 2007, TeraGrid07



Multiple Meta-Scheduler Layer Grids

Characteristics

- Multiple meta-scheduler layers in a hierarchical structure
- Resource provision in a utility fashion (provider/consumer)

Goal & Benefits

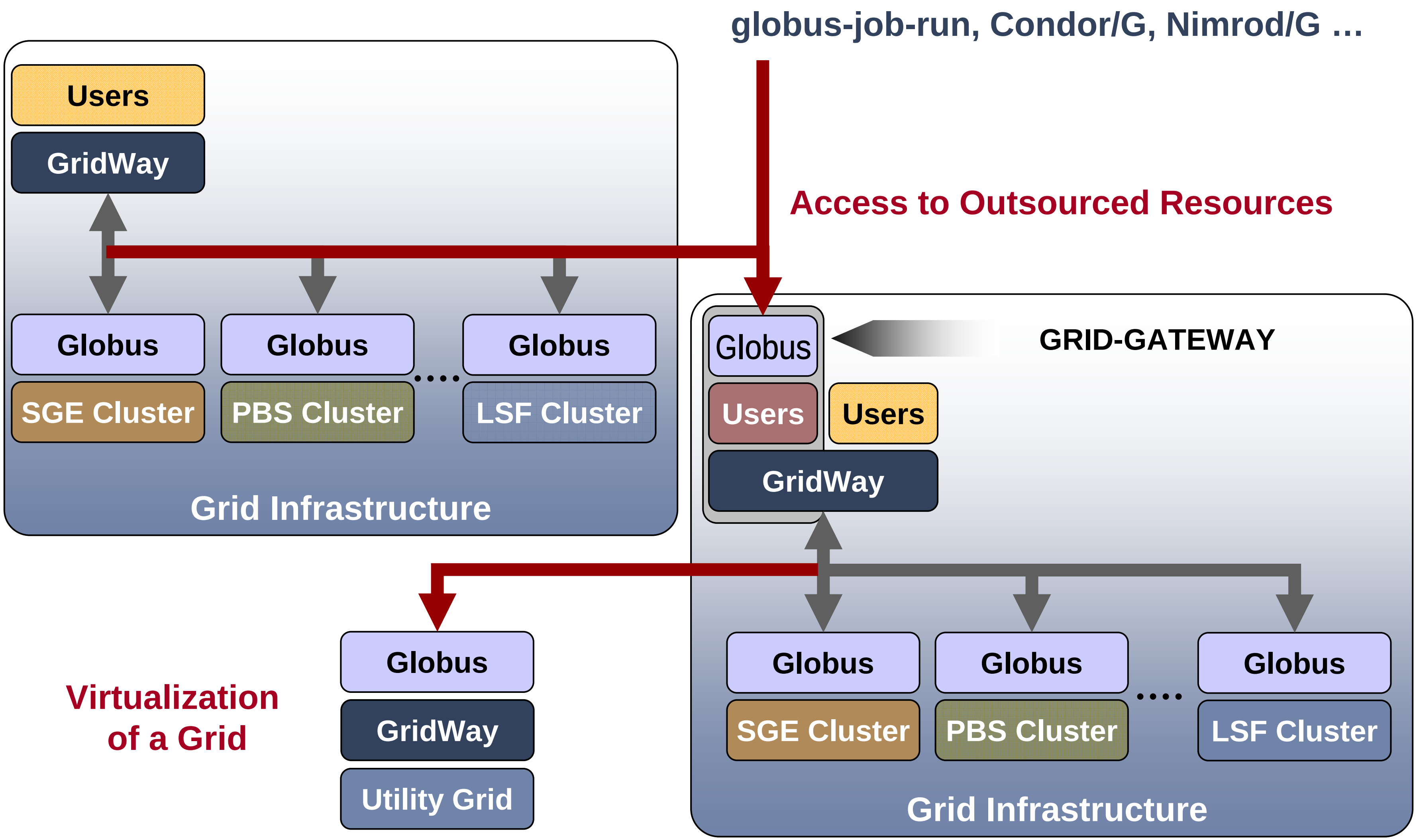
- Supply resources on-demand, making resource provision more adaptive
- Access to *unlimited* computational capacity
- Transform IT costs from fixed to variable
- Seamless integration of different Grids (The Grid)

Scheduling

- Each Grid is handled as any other resource
- Characterization of a Grid as a single resource
- Use standard interfaces to virtualize a Grid infrastructure

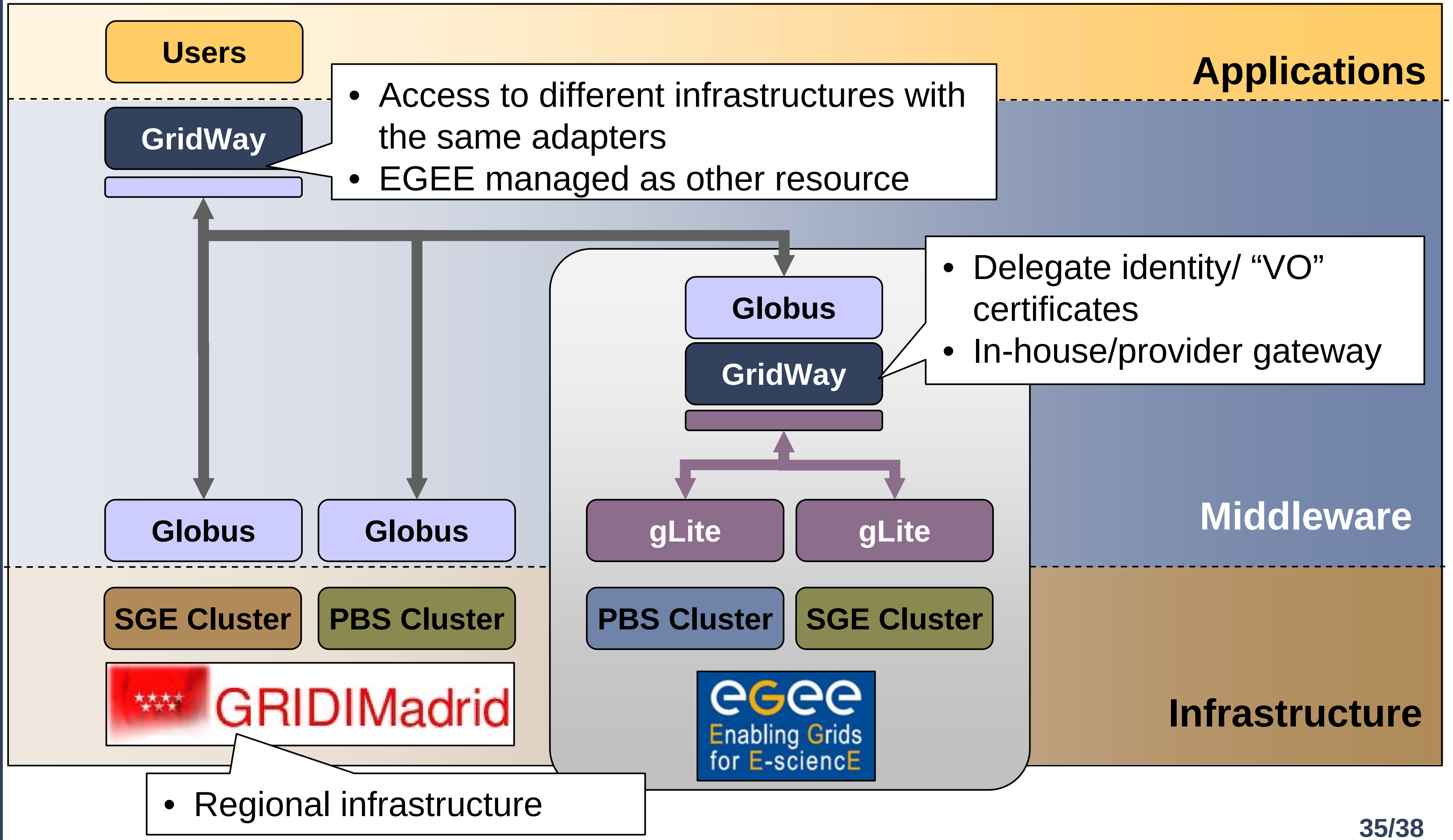
3.3. Multiple Grid Infrastructures

Deploying Multiple Meta-Scheduler Layer Grids with GridWay



3.3. Multiple Grid Infrastructures

Multiple Meta-Scheduler Layer Grids: Example



Interfaces Provided by Existing 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

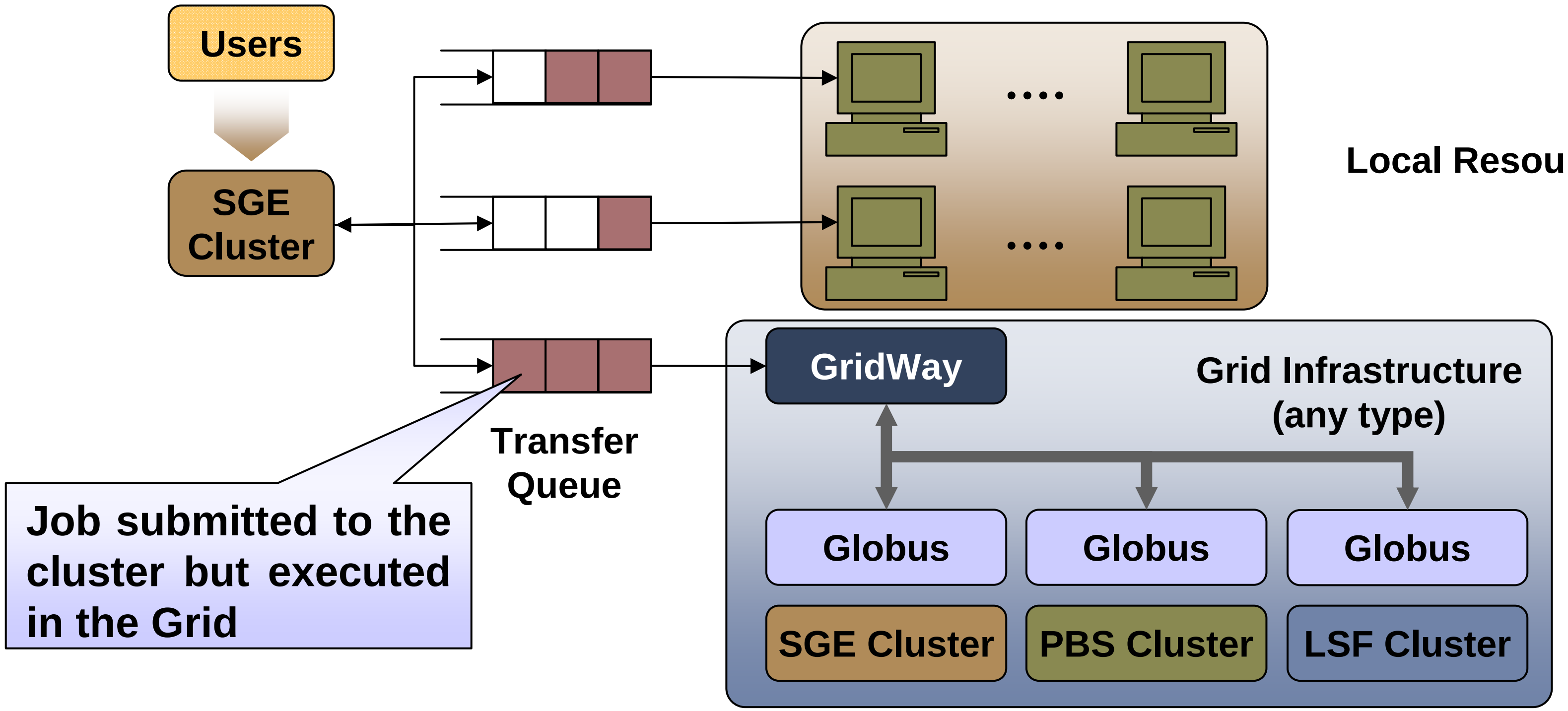
- A familiar environment to interact with a computational platform
- Some systems provide LRMS-like environment for Computational Grids
- Process still need to be adapted
- Applications would greatly benefit from standards (DRMAA)

Transfer Queues: Seamless access to the Grid

3.4. From the Cluster to the Grid

Transfer Queues: Seamless access to the 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!