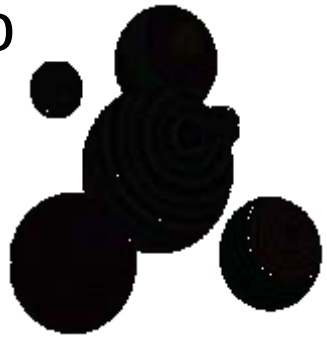
# Resource Performance Management on Computational Grids

Óscar San José
Luis Miguel Suárez
Eduardo Huedo Cuesta (huedoce@inta.es)
Rubén Santiago Montero
Ignacio Martín Llorente

**Advanced Computing Laboratory**
Centro de Astrobiología
Associated to *NASA Astrobiology Institute*
CSIC-INTA



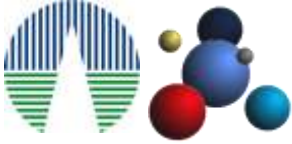


**Distributed Systems Architecture and Security Group**
Dpto. Arquitectura de Computadores y Automática
Universidad Complutense de Madrid

# Outline

- Motivation
- GRAM (*Globus Resource Allocation Manager*) Architecture
- GRPM *(Grid Resource Performance Manager*) Architecture
  - GRPM *Wrapper*
  - GRPM *Daemon*
- Integration with Grid Schedulers
- Example: High-Throughput Application
- Conclusions

# Motivation



## Globus toolkit

- Enables **secure multiple domain operation** with different **resource management systems** and **access policies.**
- Components:
  - Security Infrastructure (**GSI**)
  - Resource Management (**GRAM**)
  - Information Services (**MDS**)
  - Data Management (**GridFTP** & **Replica Management**)

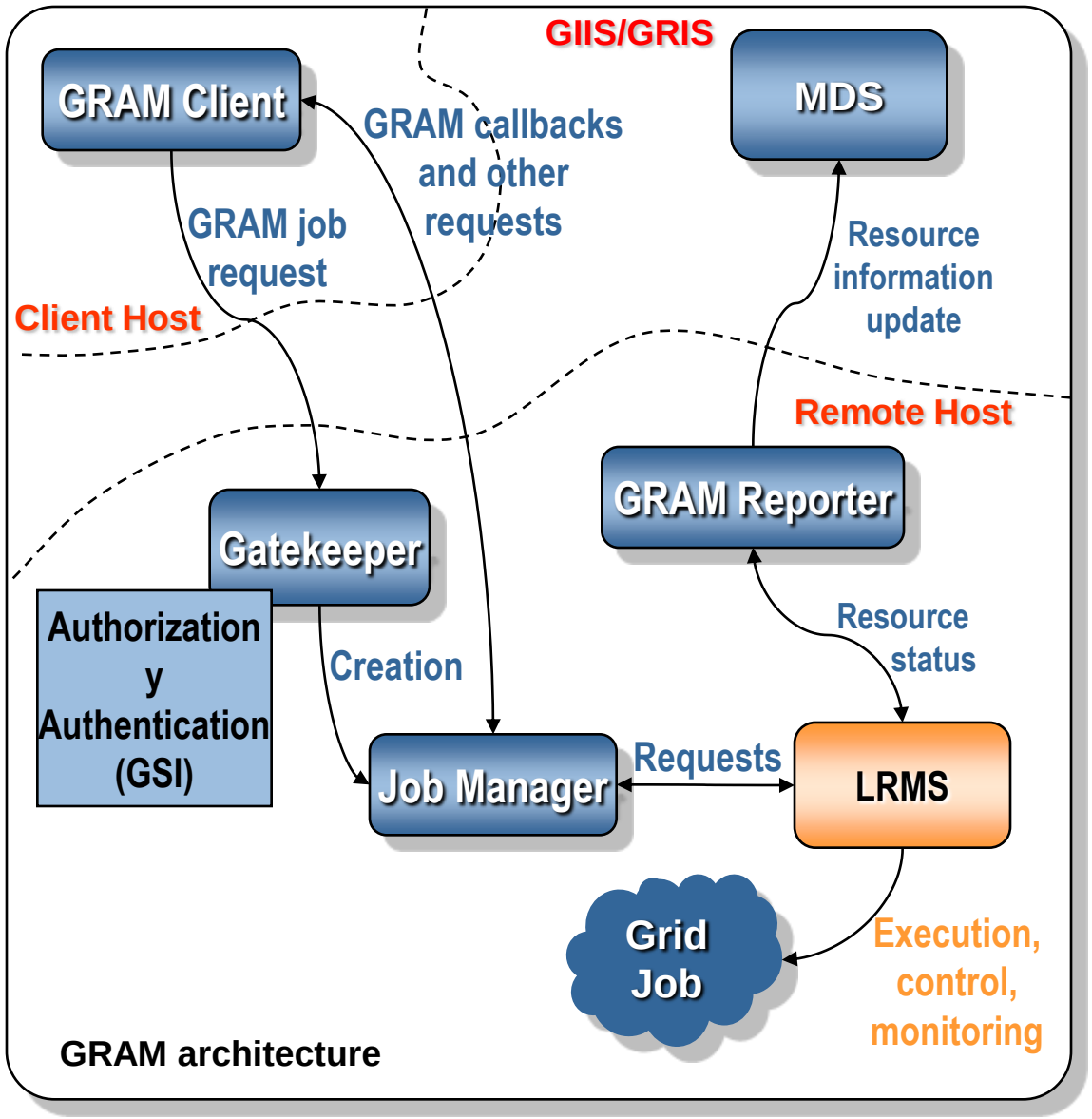
## Grid technology deployment

- Technical challenges
- **Socio-political challenges:** resource sharing policies

## Target:

Extend the Resource Management pillar with a tool that allows administrators to decide the amount of resources they are willing to devote to the Grid.

# GRAM Architecture



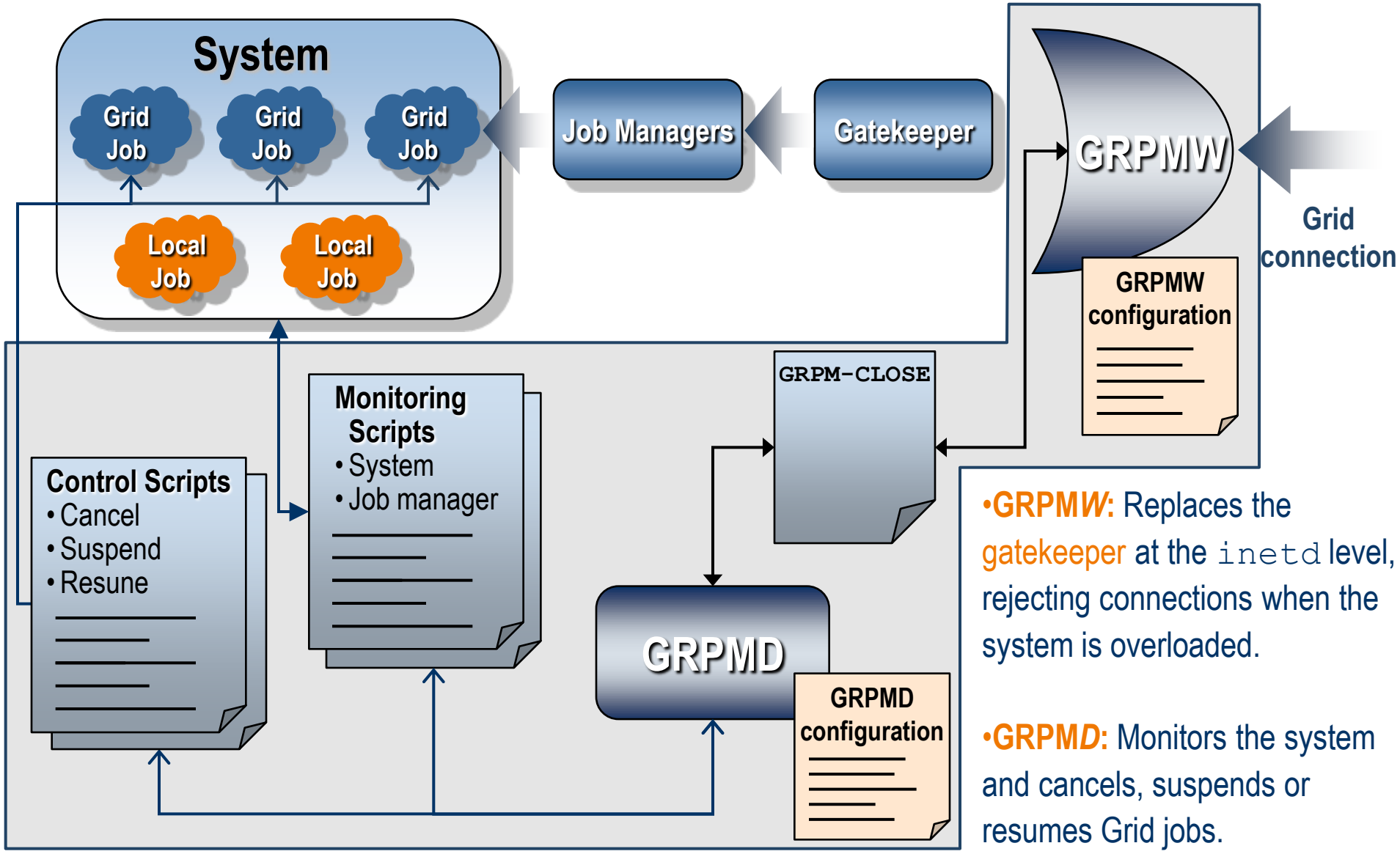

**Components:**

- **Gatekeeper:** Manages each request and creates a Job manager. Performs authentication and authorization.
- **Job manager:** Executes, controls and monitors the job following its specification (RSL). Interacts with the LRMS through scripts.
- **GRAM reporter:** Reports monitoring information to the MDS. Interacts with the LRMS through scripts.

**Drawbacks:**

- It is not possible to change **dynamically** the **authorization**.
- Once the **Grid job** is running, it behaves like any other.

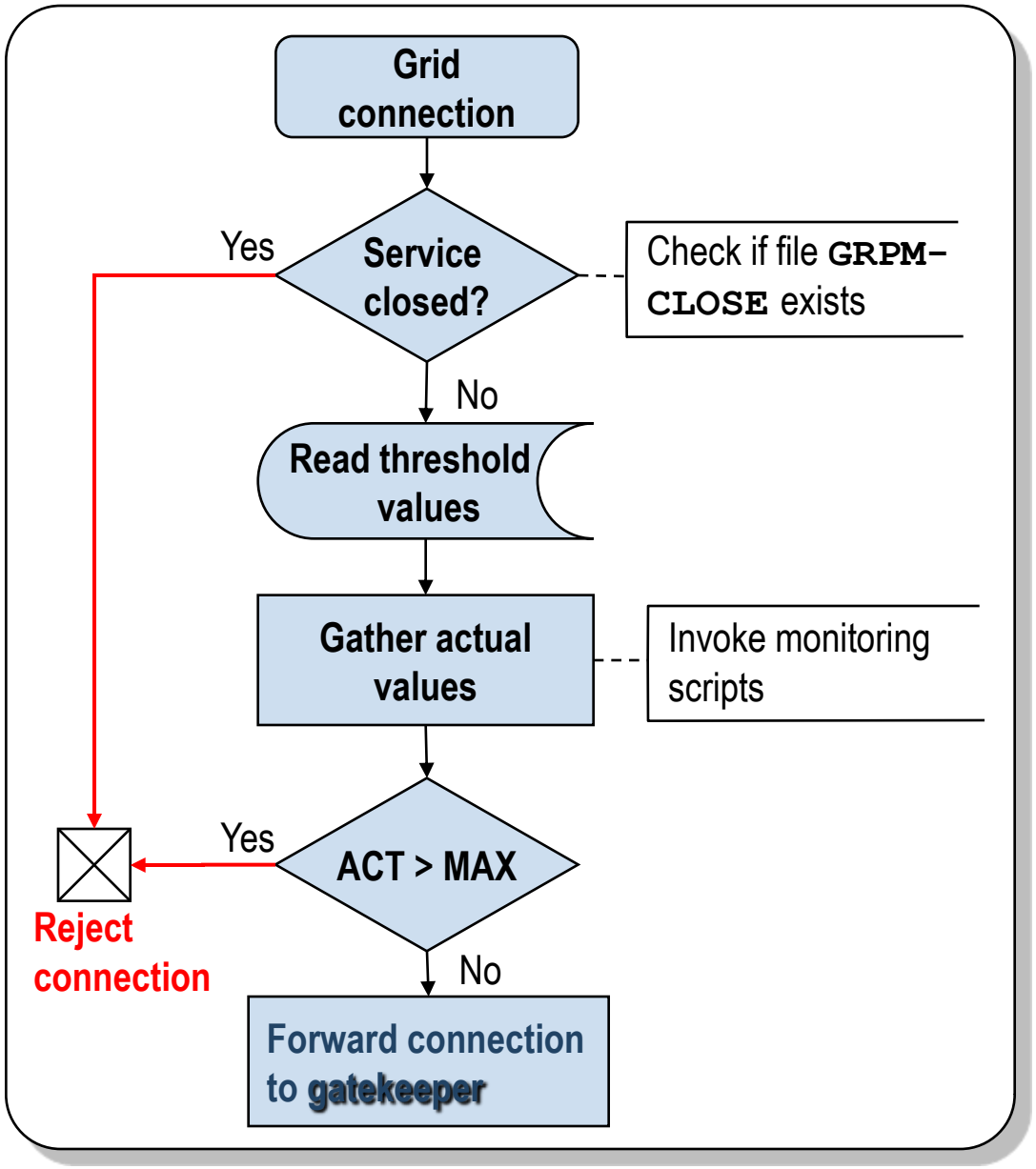
# GRPM Architecture




•**GRPM*W*:** Replaces the gatekeeper at the `inetd` level, rejecting connections when the system is overloaded.

•**GRPM*D*:** Monitors the system and cancels, suspends or resumes Grid jobs.

# GRPM *Wrapper*



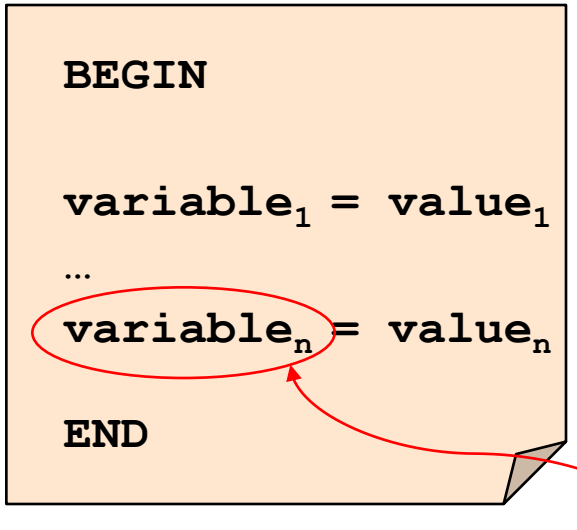

**Components:**

- Interface to retrieve information from the system (*GRAM reporter* scripts).
- Configuration variables (*thresholds*):
  - **MAXUSERS**
  - **MAXGRIDUSERS**
  - **MAXJOBS**
  - **MAXGRIDJOBS**
- It allows the assignment of different policies to different *job managers* (fork, PBS…)

# GRPM *Daemon*

**Components:**

- Interface to retrieve information from the system (*GRAM reporter* scripts):

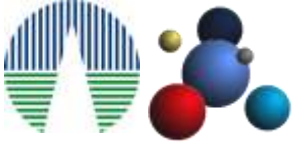



- Performance expressions: **f(variable$_1$,…,variable$_n$)** ∈ {**TRUE,FALSE**}
  - **CANCEL**
  - **SUSPEND**

  } Creation of **GRPM-CLOSE** file

  - **RESUME** — Deletion of **GRPM-CLOSE** file
- Interface to interact with the LRMS (*Job manager* scripts).

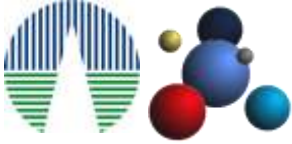
# Grid Schedulers



## Grid Schedulers (Super-Schedulers, Meta-Schedulers...)

- Resource scheduling across **multiple administration domains**: resource discovery and selection; and job preparation, submission, monitoring, migration and termination.
- Interaction with different scheduling steps:
  - **Resource Discovery**
    - The Grid scheduler should perform an authorization to guarantee user access, so it could detect a rejection.
  - **Job Submission**
    - The Grid scheduler is notified about submission failures, so it could detect a rejection.
  - **Job Monitoring**
    - The Grid scheduler is notified about job state changes, so it could detect a suspension.
  - **Job Termination**
    - The Grid scheduler usually executes jobs through *wrappers* to capture their exit codes, so it could detect a cancellation.

# Example: High-Throughput Application



## Testbed description:

| Host | Model | Hz | OS | Memory | Nodes | GRAM |
|---|---|---|---|---|---|---|
| ursa | Sun Blade 100 | 500Mhz | Solaris 8 | 256MB | 1 | fork |
| draco | Sun Ultra 1 | 167Mhz | Solaris 8 | 128MB | 1 | fork |
| **pegasus** | **Intel Pentium 4** | **2.4 Ghz** | **Linux 2.4** | **1GB** | **1** | **fork** |
| babieca | Alpha DS10 | 466 Mhz | Linux 2.2 | 1GB | 4 | PBS |
| solea | Sun Enterprise 250 | 296 Mhz | Solaris 8 | 256MB | 2 | fork |

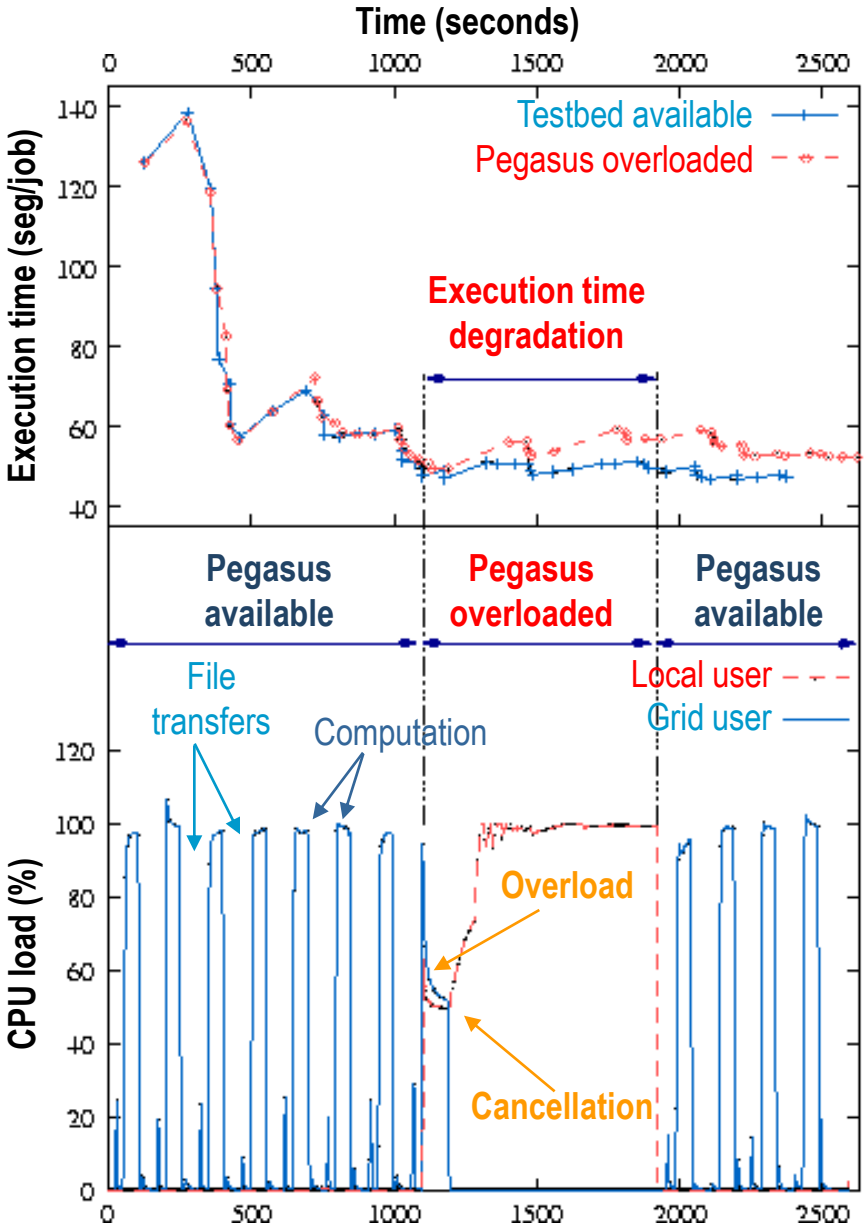
## Experiment:

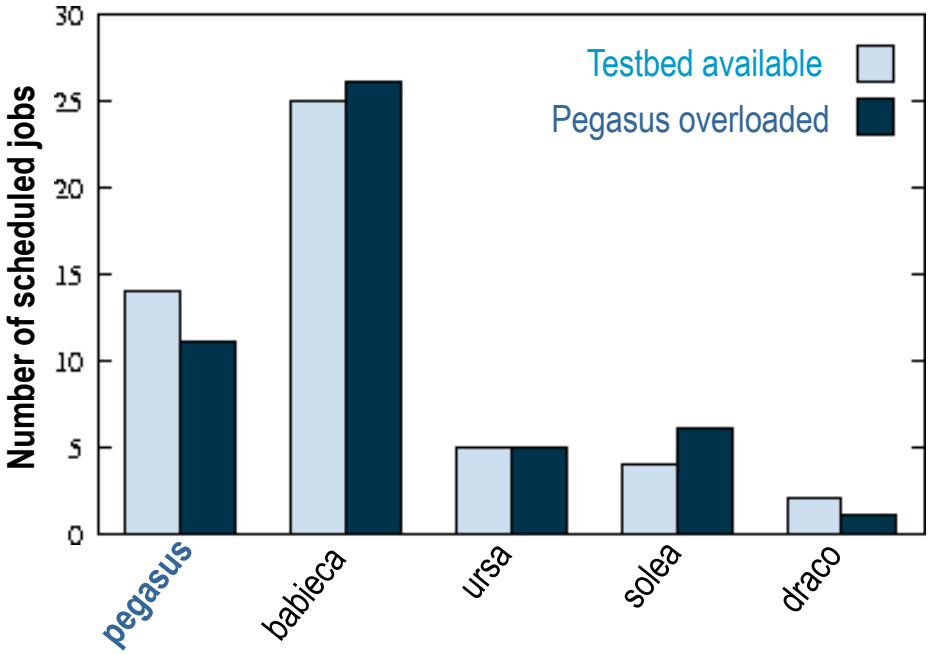
- Preserve the performance of a workstation (**pegasus**) following:

```
MAX_GRID_USERS = 1

CANCEL = local_cpu_load > 25

RESUME = local_cpu_load < 15
```

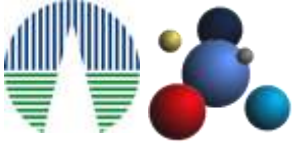
- Execution of a high-throughput application (CFD) consisting of 50 tasks ($Re=10^2 – 10^4$) using the **Grid*Way*** tool.

# Example: High-Throughput Application



# Conclusions



**Grid Resource Performance Manager**

- Development of a performance manager for Grid resources
- It does not interfere with Grid schedulers
- It protects resource owners from:
  - *selfish* schedulers
  - non-existent or obsolete monitoring information

**Characteristics:**

- **Portable**
- **Homogeneous**
- **Decentralized**
- **Easily installable and deployable**