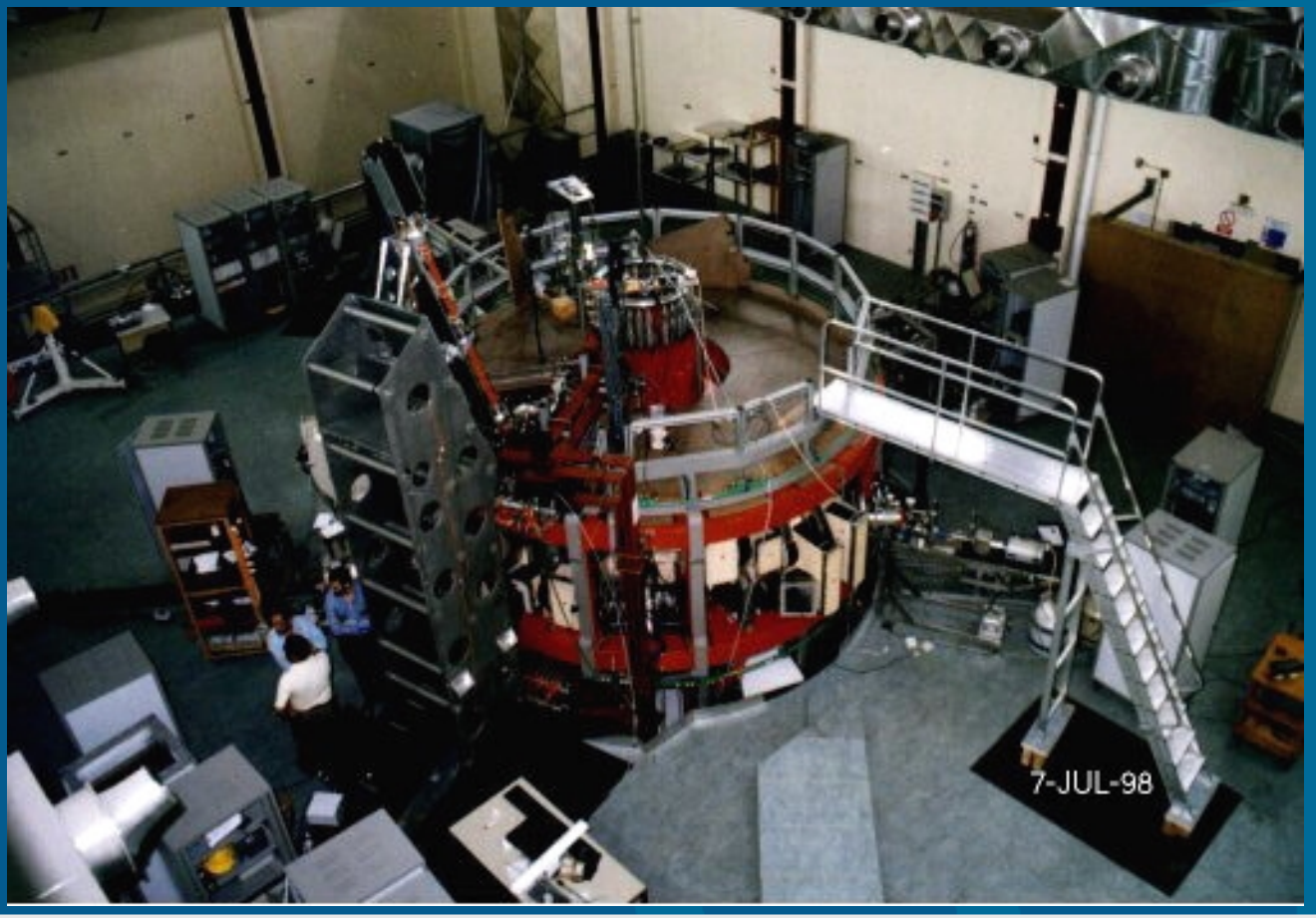



# Massive Ray Tracing in Fusion Plasmas on EGEE

J.L. Vázquez-Poletti, E. Huedo, R.S. Montero and I.M. Llorente

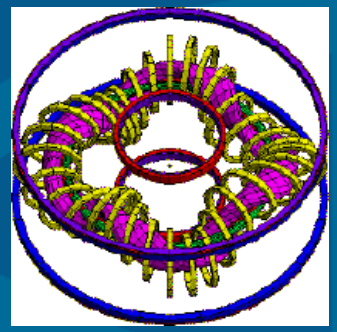
Distributed Systems Architecture Group
Universidad Complutense de Madrid (Spain)

EGEE User Forum
1st - 3rd March, CERN (Geneva)

# What are we going to see?

- MA-RA-TRA/G: a computational view
- Execution using the LCG-2 Resource Broker
- Execution using the GridWay Meta-Scheduler
- Comparison
- Conclusions
- “Our two cents”

# MA-RA-TRA/G: a computational view

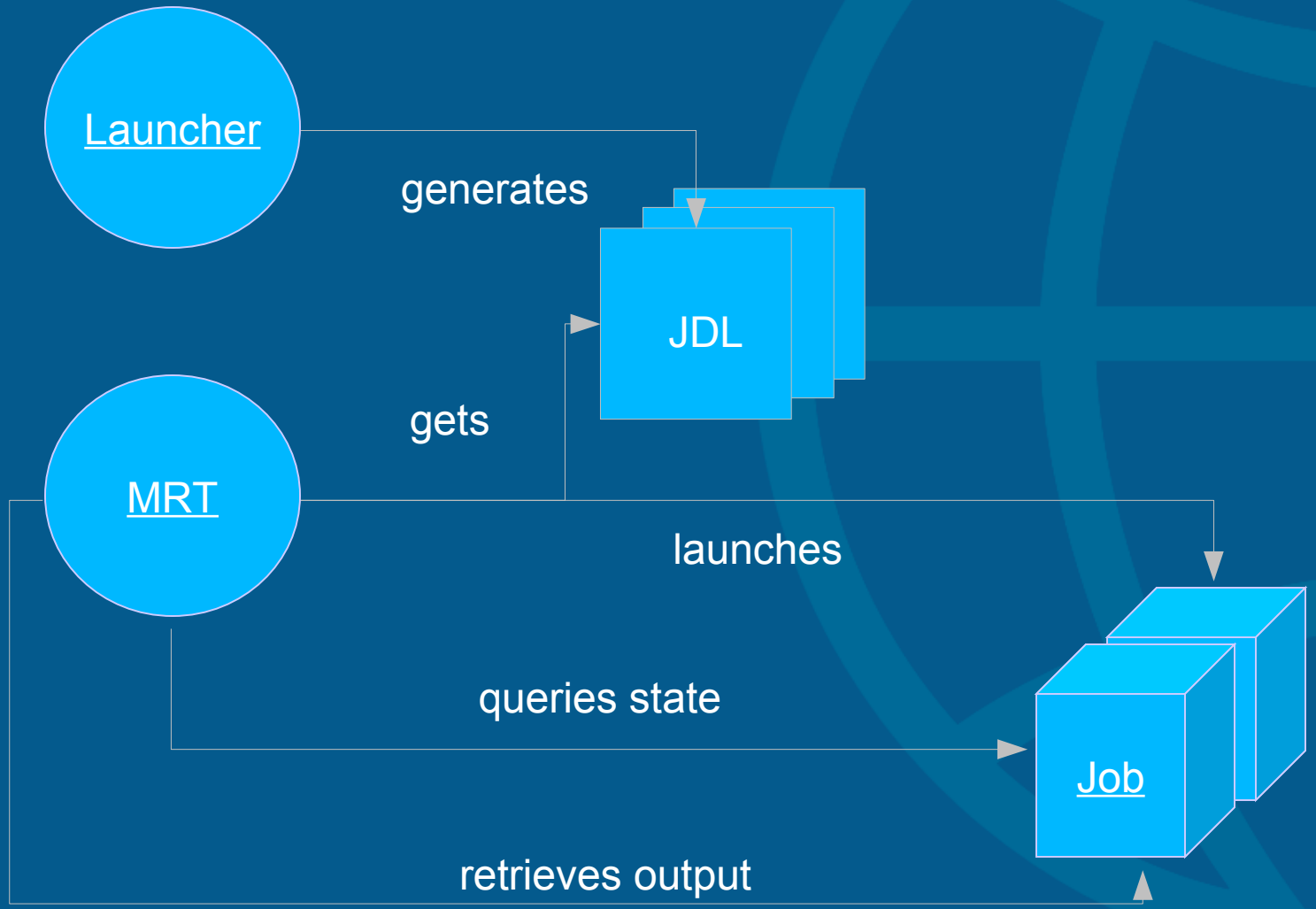


- MA-RA-TRA/G activity: “Massive Ray Tracing in Fusion Plasmas on Grids”
- Application profile:
  - Sizes
    - Executable (Truba) – 1.8 MB
    - Input files – 70 KB
    - Output files – about 459 KB
  - Execution Time – about 26 minutes
    - Pentium 4 (3.2 GHz)
  - 1 execution = 1 ray traced

# Execution using the LCG-2 Resource Broker

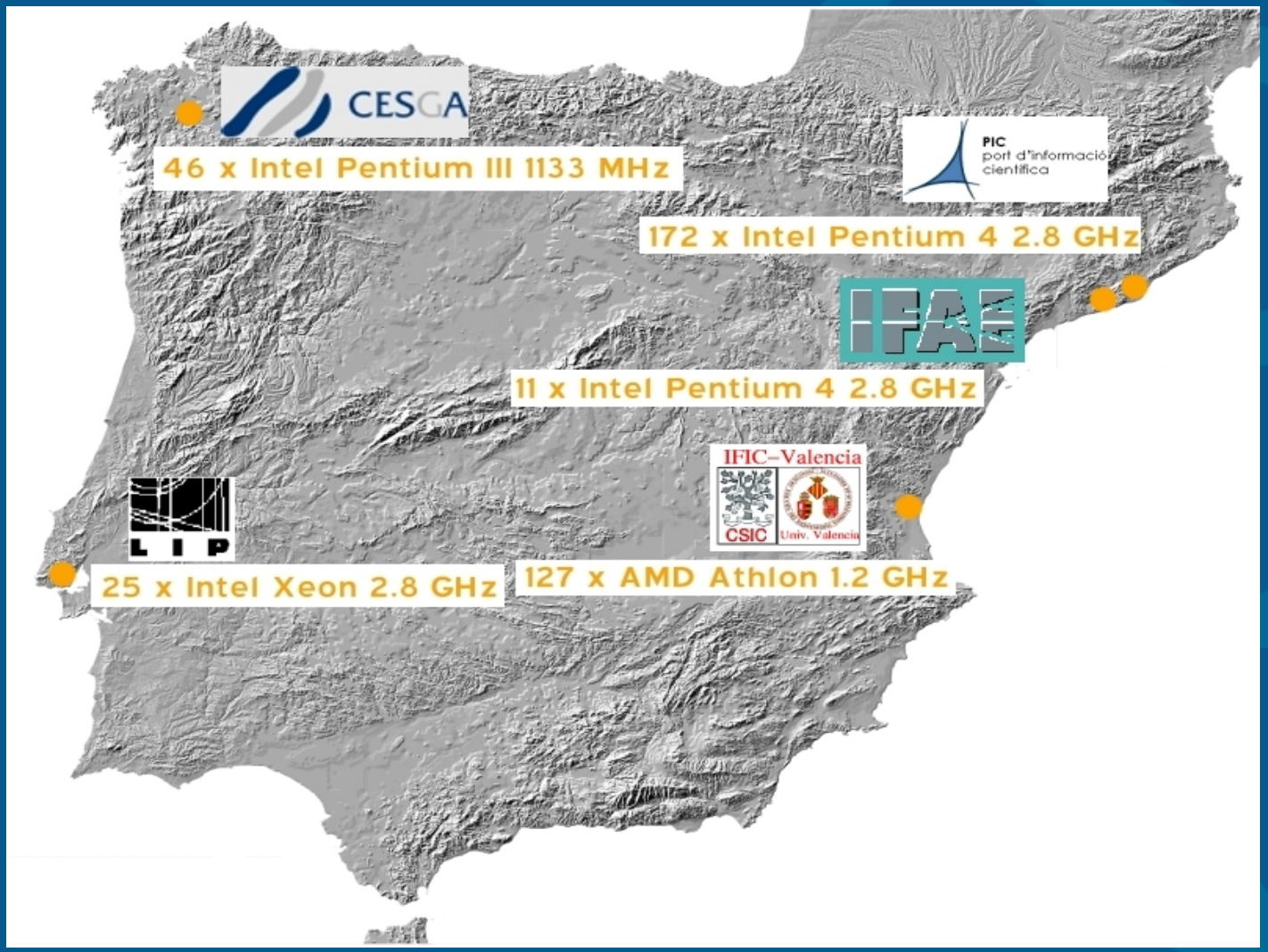

- lcg2.1.9 User Interface C++ API
- 1 job = 1 ray
- Procedure:
  - Launcher script
    - Generates JDL files
  - Framework over LCG-2
    - Launches them simultaneously
    - Queries each job's state periodically
    - Retrieves each job's output (Sandbox)

# Execution using the LCG-2 Resource Broker

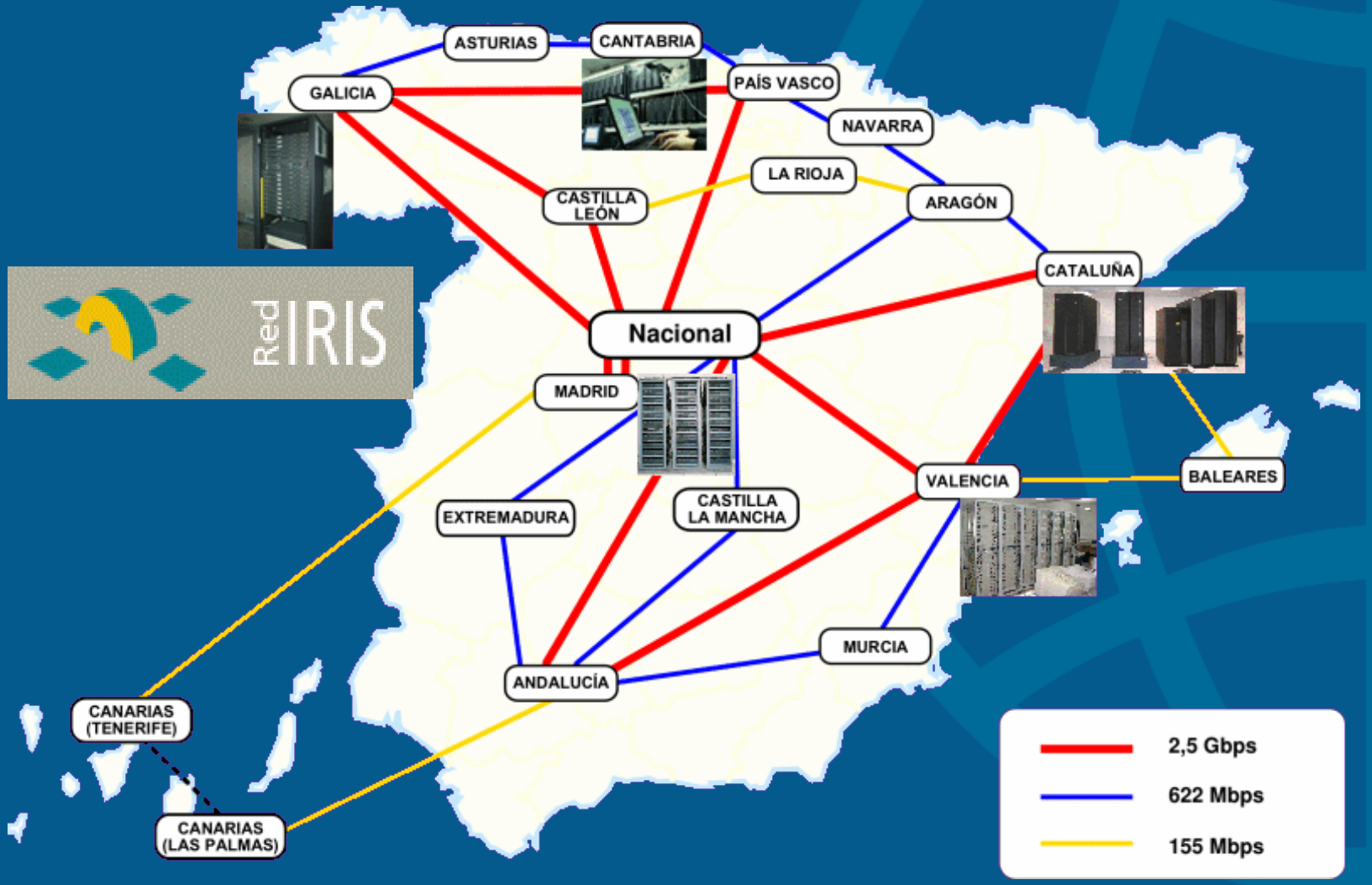


# Execution using the LCG-2 Resource Broker

SWETEST VO



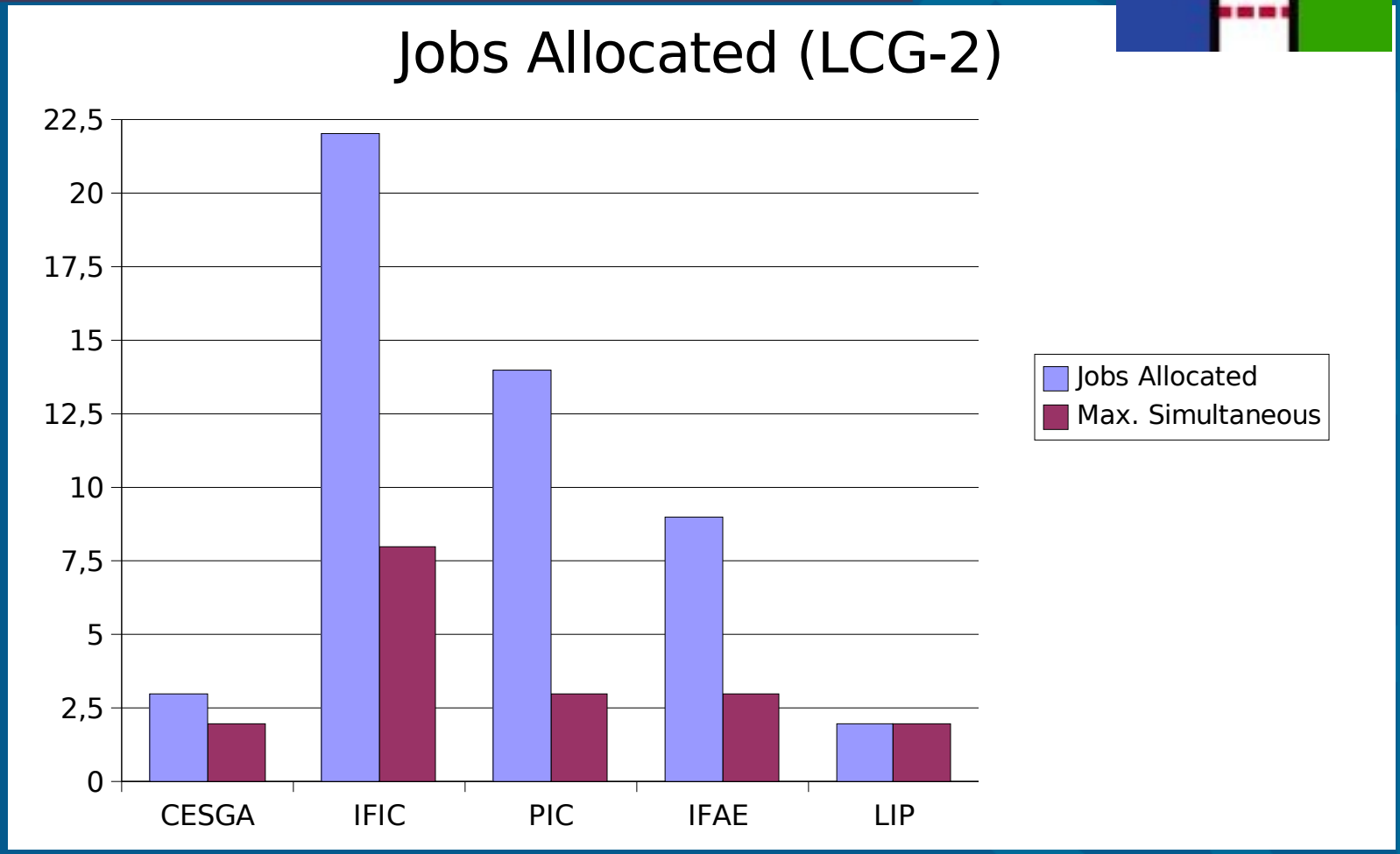
# Execution using the LCG-2 Resource Broker



# Execution using the LCG-2 Resource Broker


- Total Time: **220** minutes (**3.67** hours)
- Execution Time:
  - Average: **30.33** minutes
  - Std. Deviation: **11.38** minutes
- Transfer Time:
  - Average: **0.42** minutes
  - Std. Deviation: **0.06** minutes
- Avg. Productivity: **13.36** Jobs/hour
- Avg. Overhead: **1.82** minutes/job

# Execution using the LCG-2 Resource Broker




CPU normalized to reference value of 1000 SepctInt2000 (Pentium 4 2.8 GHz)

42.86

# Execution using the LCG-2 Resource Broker


- As in the real world, some jobs failed
  - Jobs affected: 31
  - Max resubmissions/job: 1
- Problems encountered:
  - LCG-2 Infrastructure:
    - Lack of opportunistic migration
    - No slowdown detection
    - Jobs assigned to busy resources
  - The API itself:
    - Submitting more than 80 jobs in a Collection (empirically)
    - Not a standard

# Execution using the GridWay Meta-Scheduler


- Open-source Meta-Scheduling framework
- Works on top of Globus services
- Performs:
  - Job execution management
  - Resource brokering
- Allows unattended, reliable and efficient execution of:
  - single jobs, array jobs, complex jobs
  - on heterogeneous, dynamic and loosely-coupled grids

# Execution using the GridWay Meta-Scheduler


- Works transparently to the end user
- Adapts job execution to changing Grid conditions
  - Fault recovery
  - Dynamic scheduling
  - Migration on-request
- Scheduling using Information System (GLUE schema) from LCG-2
- Stands on the client side

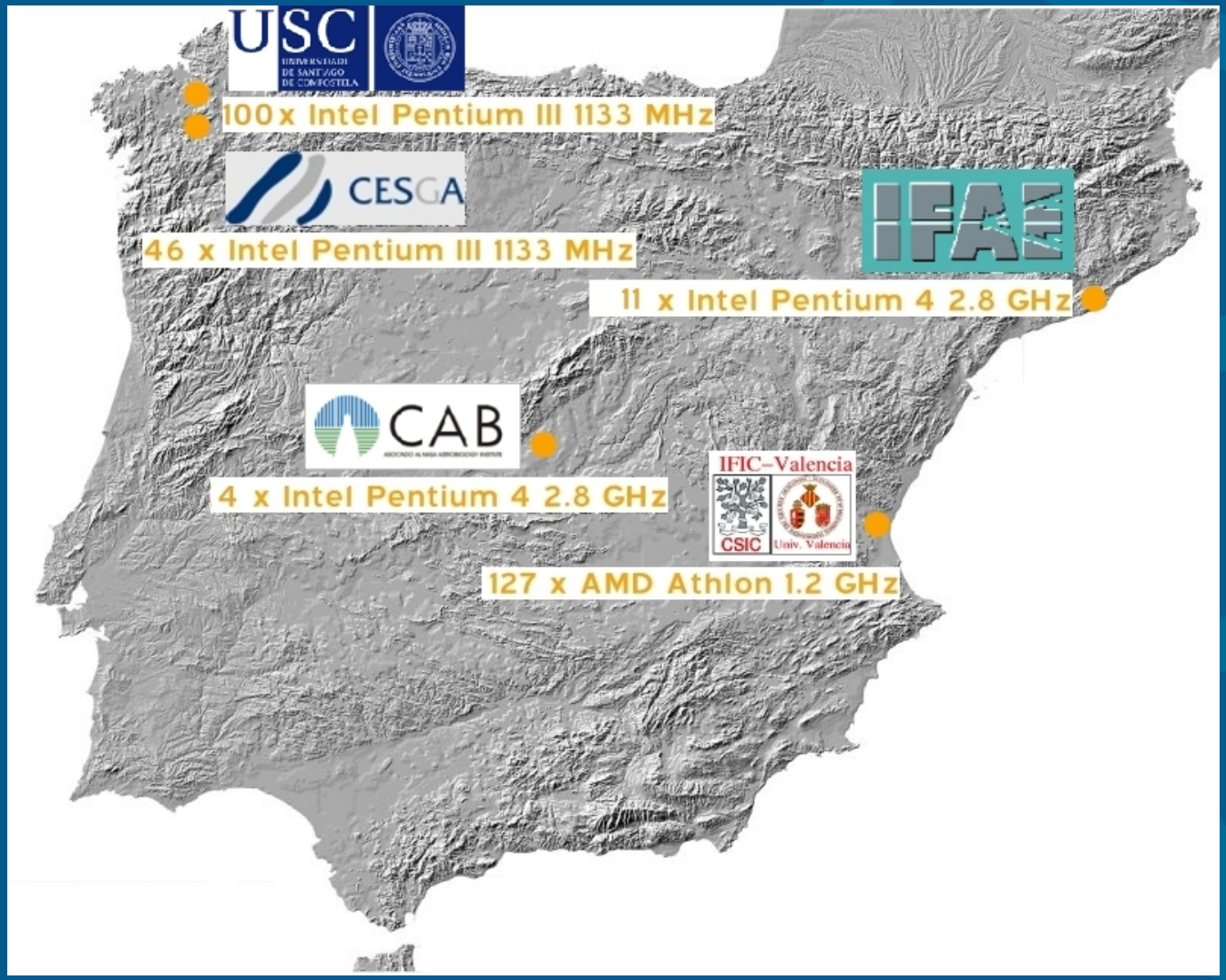
# Execution using the GridWay Meta-Scheduler


- Execution as seen by GridWay:
  - *Prolog*: prepares remote system
    - Creates directory
    - Transfers input files and executable
  - *Wrapper*: executes job and gets exit code
  - *Epilog*: finalizes remote system
    - Transfers output files
    - Cleans up directory

# Execution using the GridWay Meta-Scheduler

www.gridway.org

## SWETEST VO

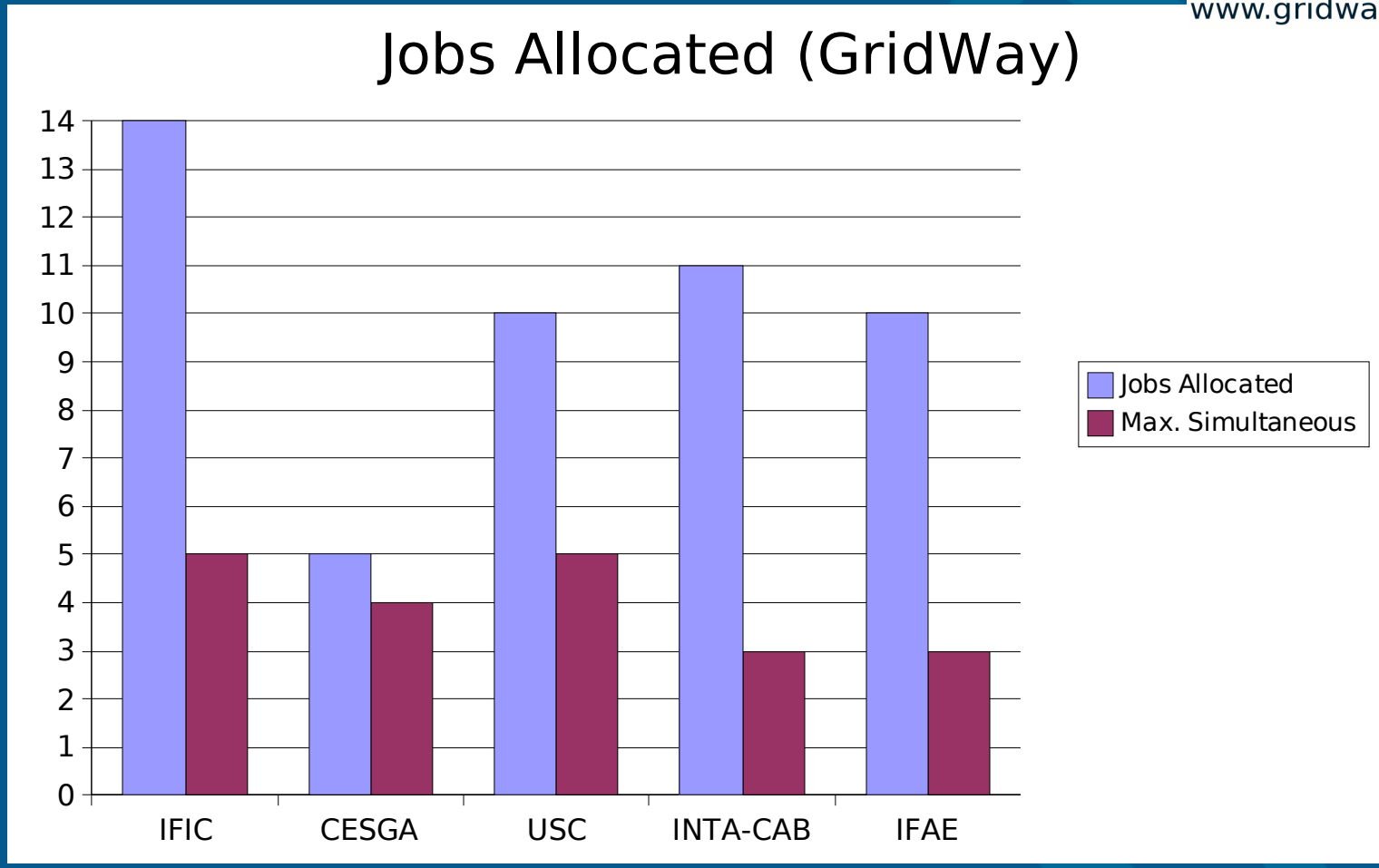


# Execution using the GridWay Meta-Scheduler


- Total Time: **123.43** minutes (**2.06** hours)
- Execution Time:
  - Average: **36.8** minutes
  - Std. Deviation: **16.23** minutes
- Transfer Time:
  - Average: **0.87** minutes
  - Std. Deviation: **0.51** minutes
- Avg. Productivity: **23.82** Jobs/hour
- Avg. Overhead: **0.52** minutes/job

# Execution using the GridWay Meta-Scheduler

www.gridway.org

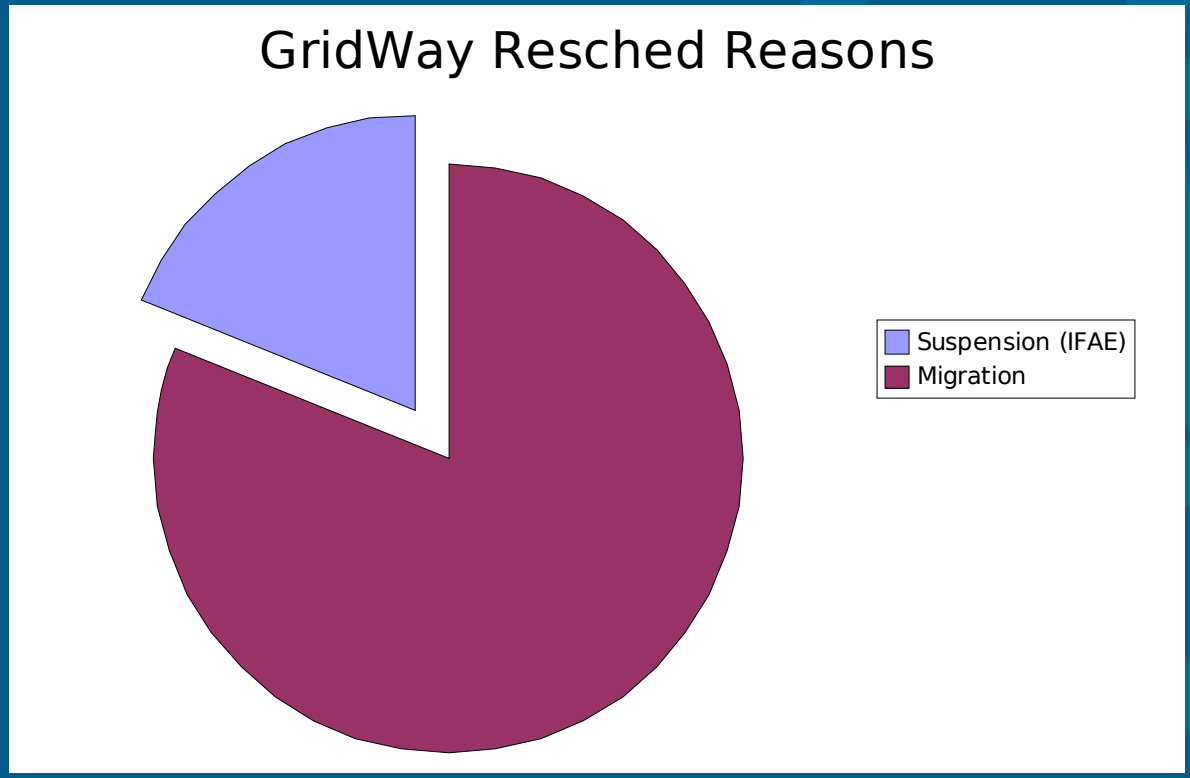



CPU normalized to reference value of 1000 SepctInt2000 (Pentium 4 2.8 GHz)

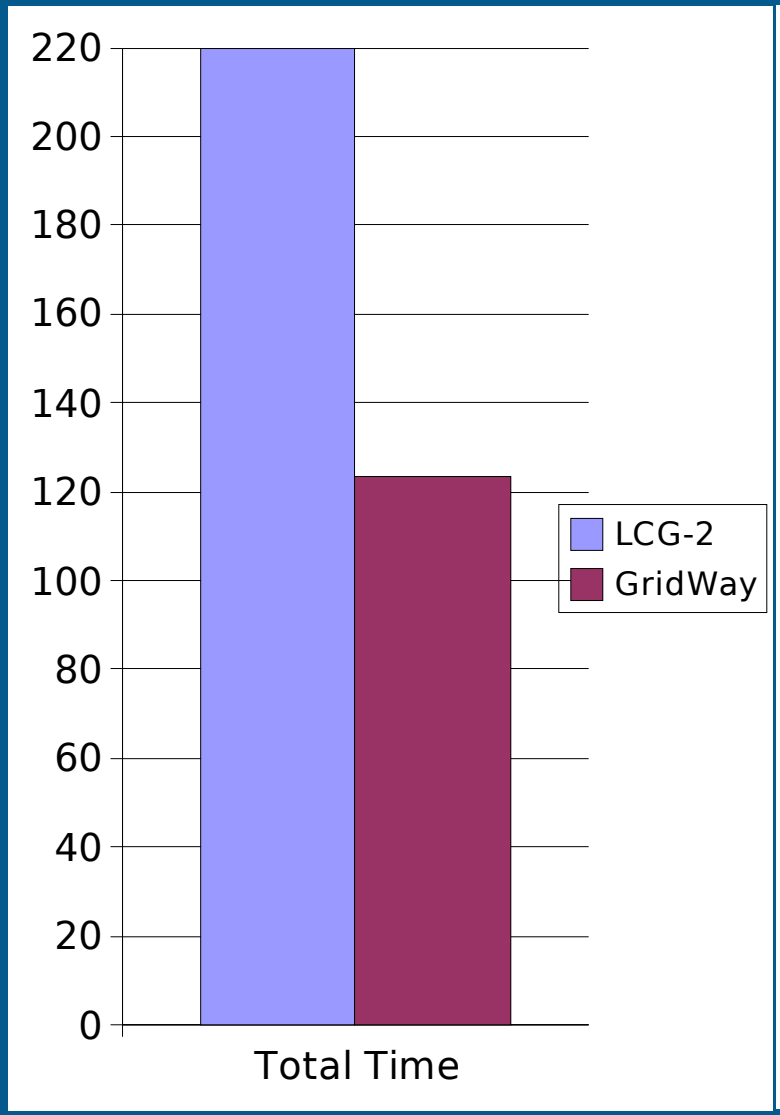
48.54

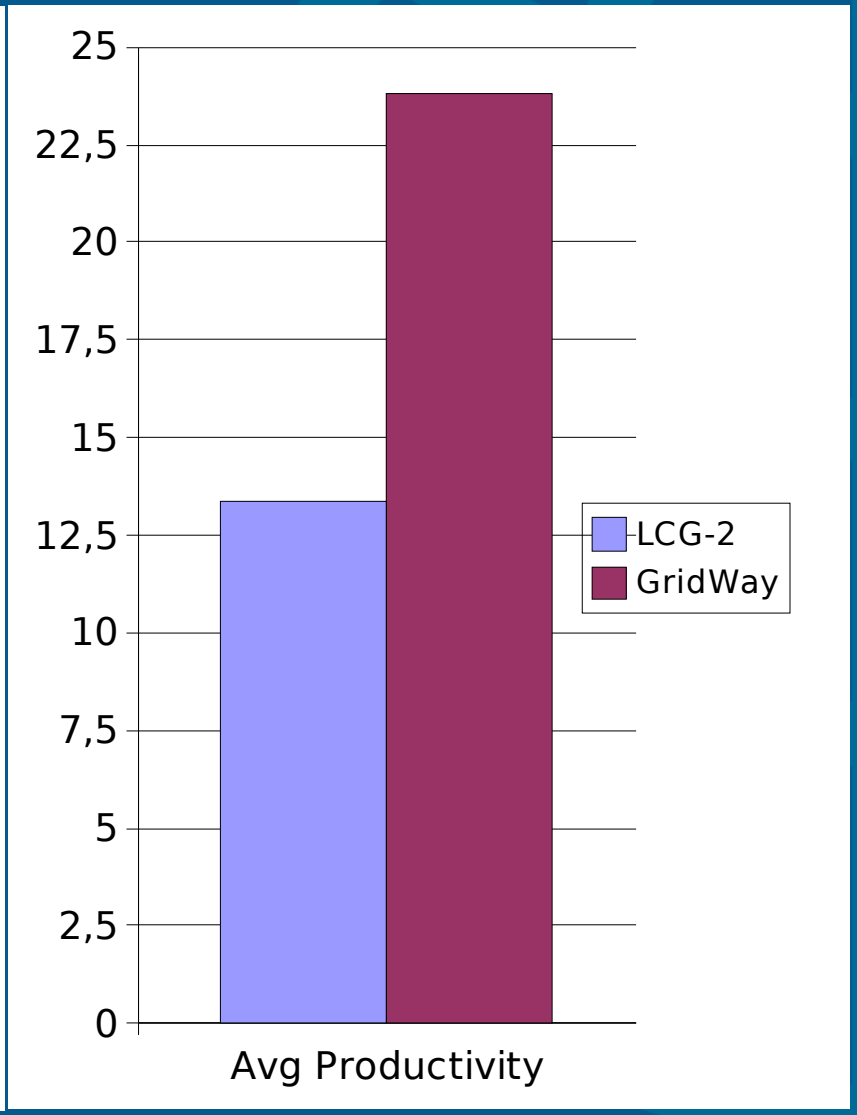
# Execution using the GridWay Meta-Scheduler

- Also with GridWay, some jobs failed: 1
- Reschedules: 21
  - Max./job: 4

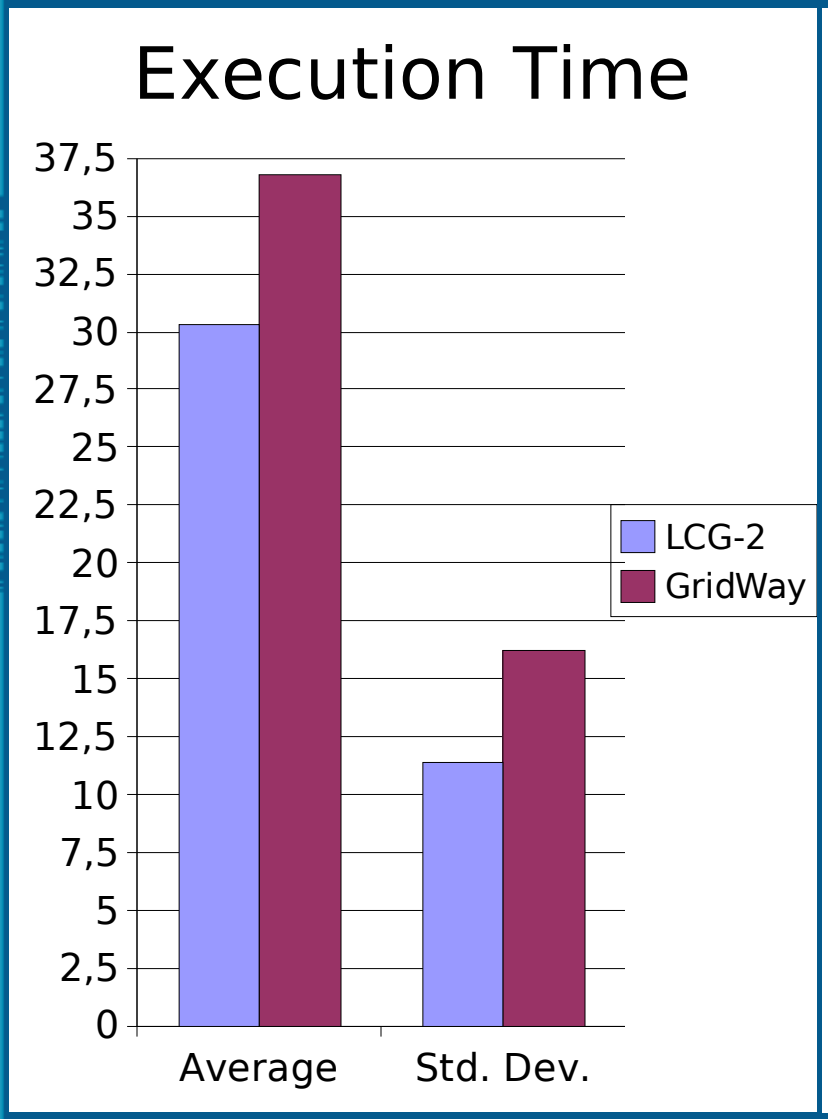


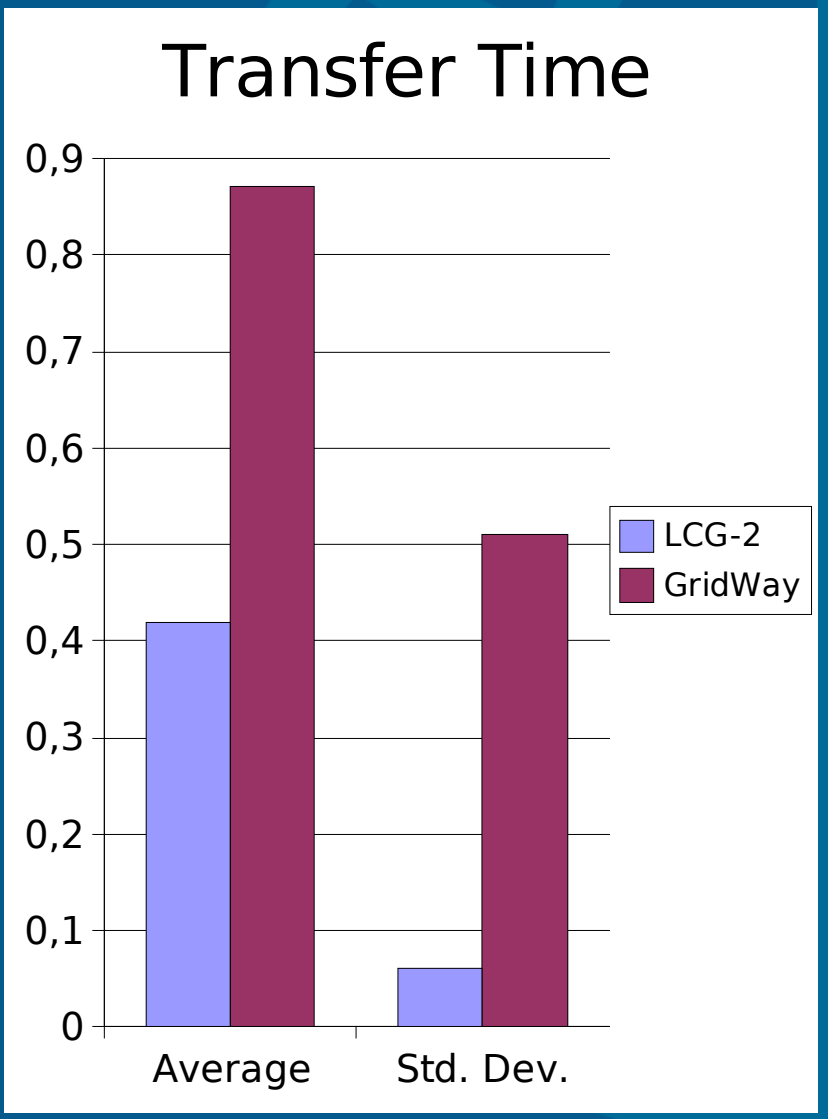
# Comparison


220
200
180
160
140
120
100
80
60
40
20
0
LCG-2
GridWay
Total Time

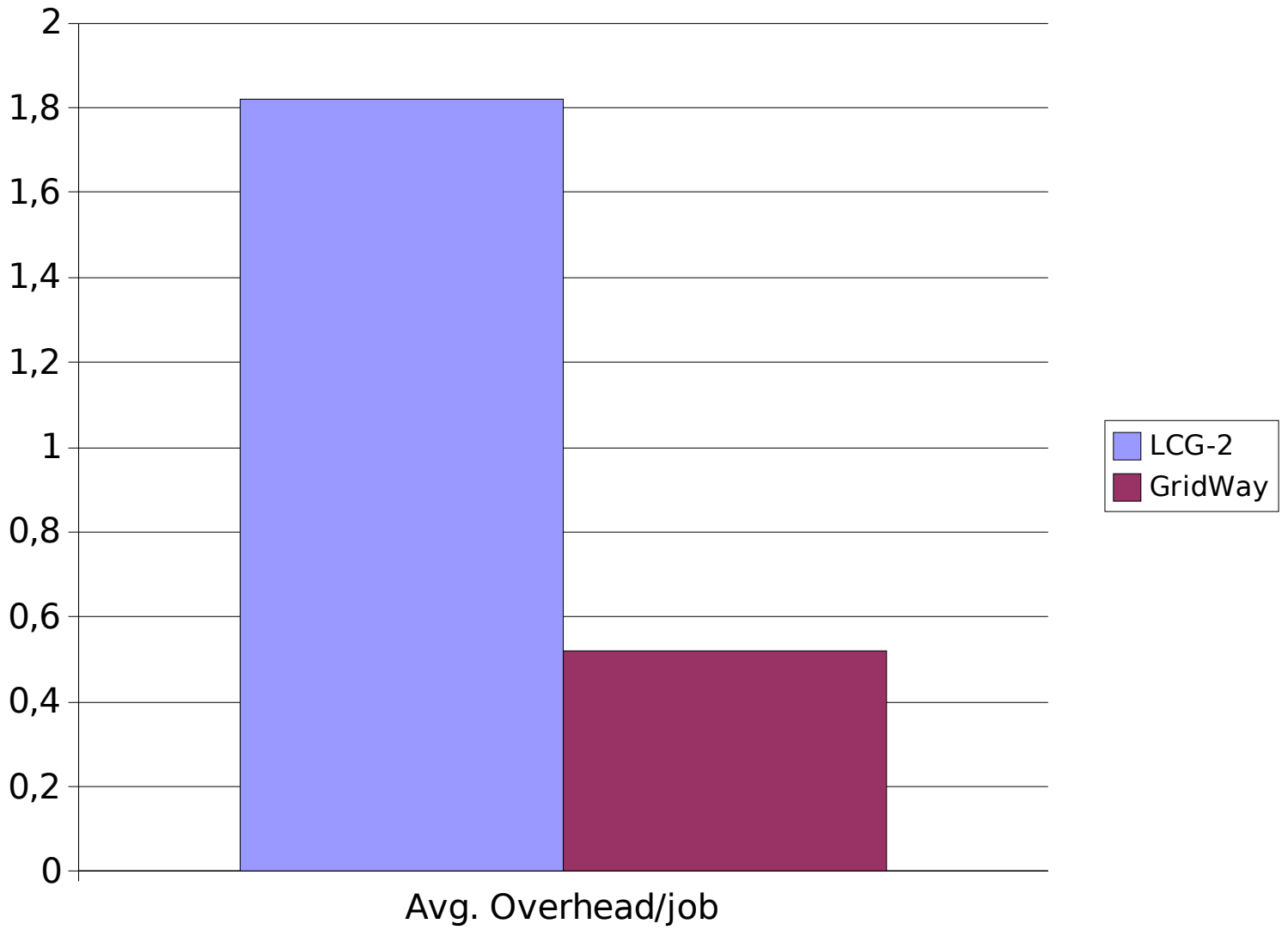

25
22,5
20
17,5
15
12,5
10
7,5
5
2,5
0
LCG-2
GridWay
Avg Productivity

# Comparison

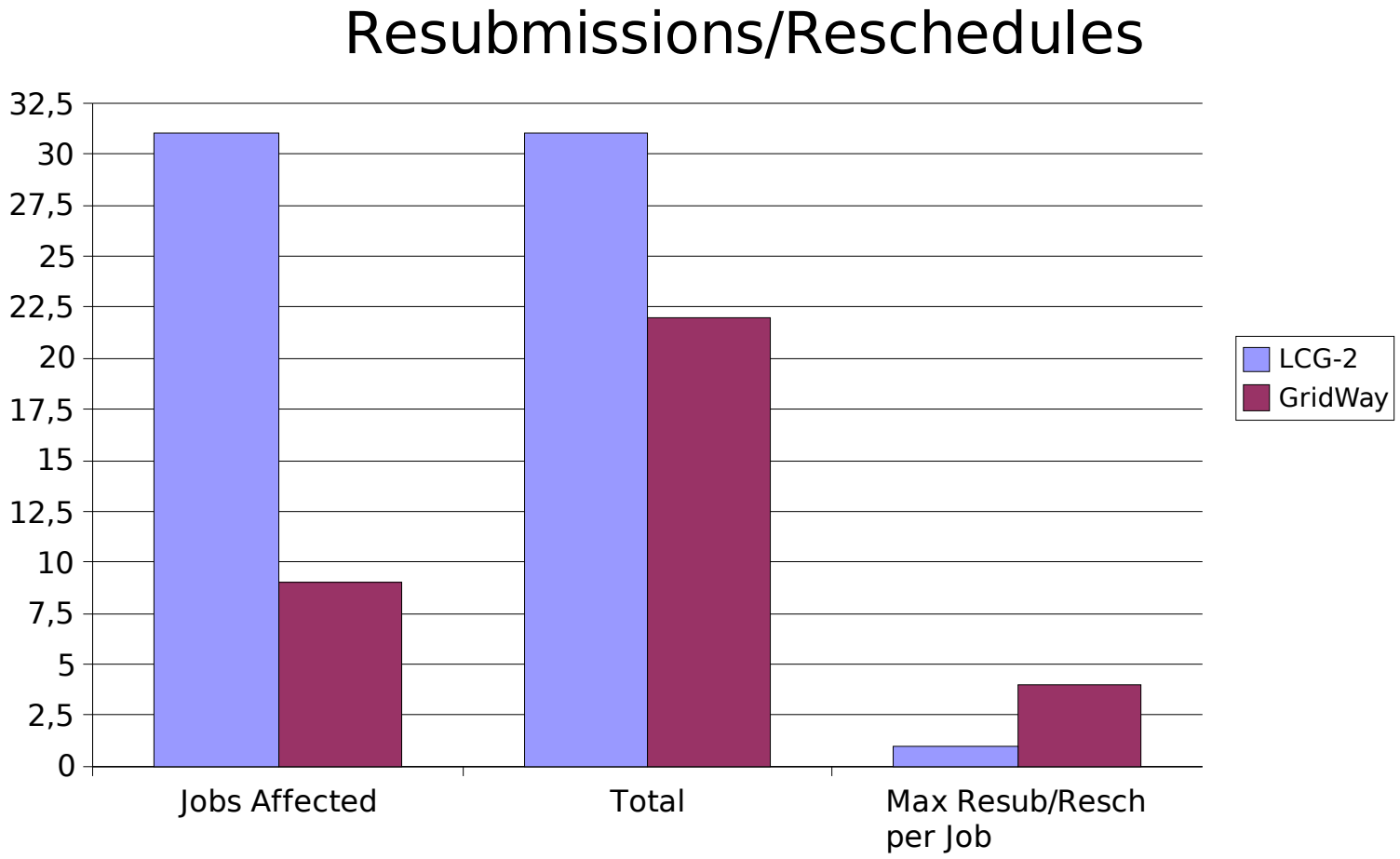

Execution Time
37,5
35
32,5
30
27,5
25
22,5
20
17,5
15
12,5
10
7,5
5
2,5
0
Average
Std. Dev.
LCG-2
GridWay


Transfer Time
0,9
0,8
0,7
0,6
0,5
0,4
0,3
0,2
0,1
0
Average
Std. Dev.
LCG-2
GridWay

# Comparison

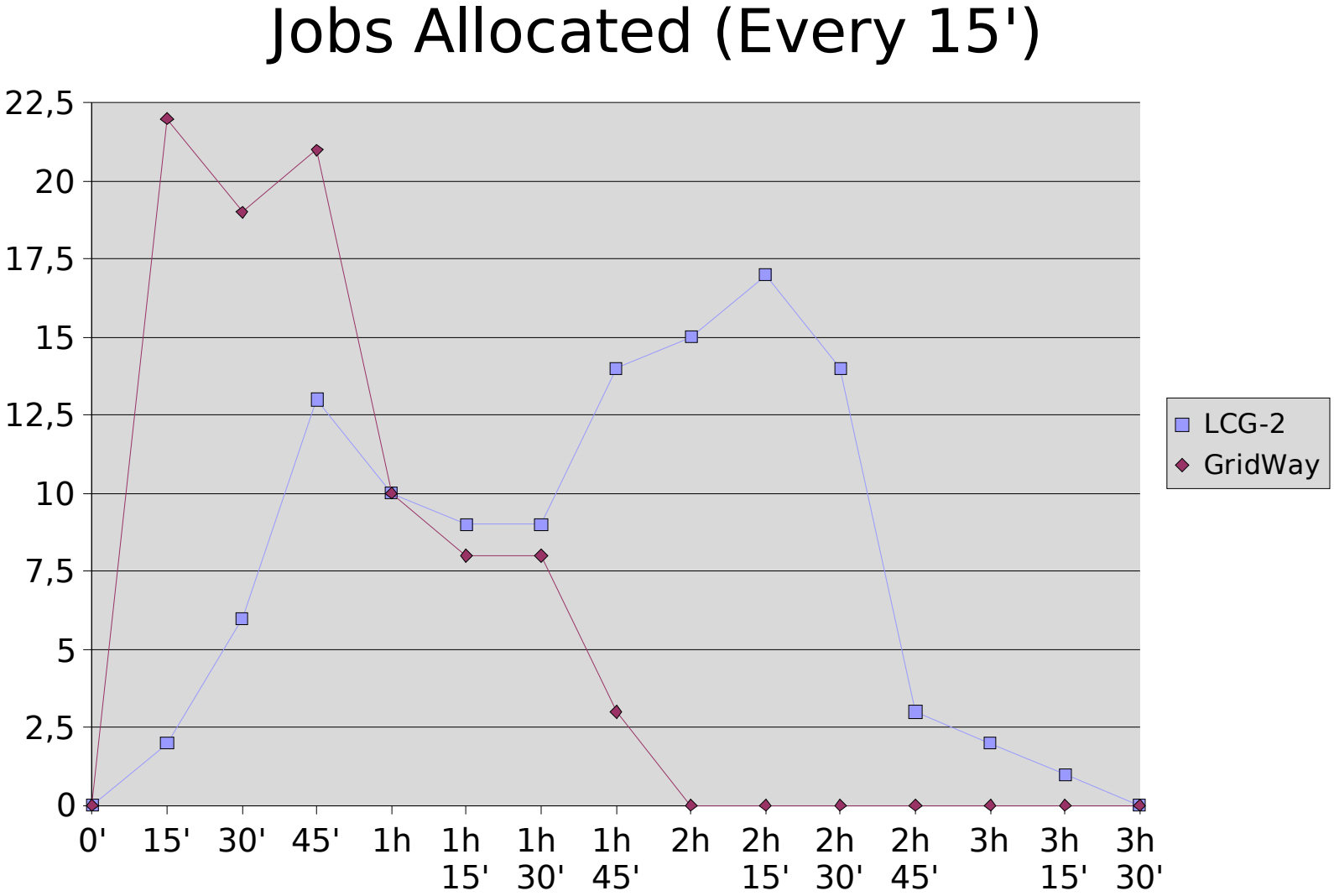


# Comparison

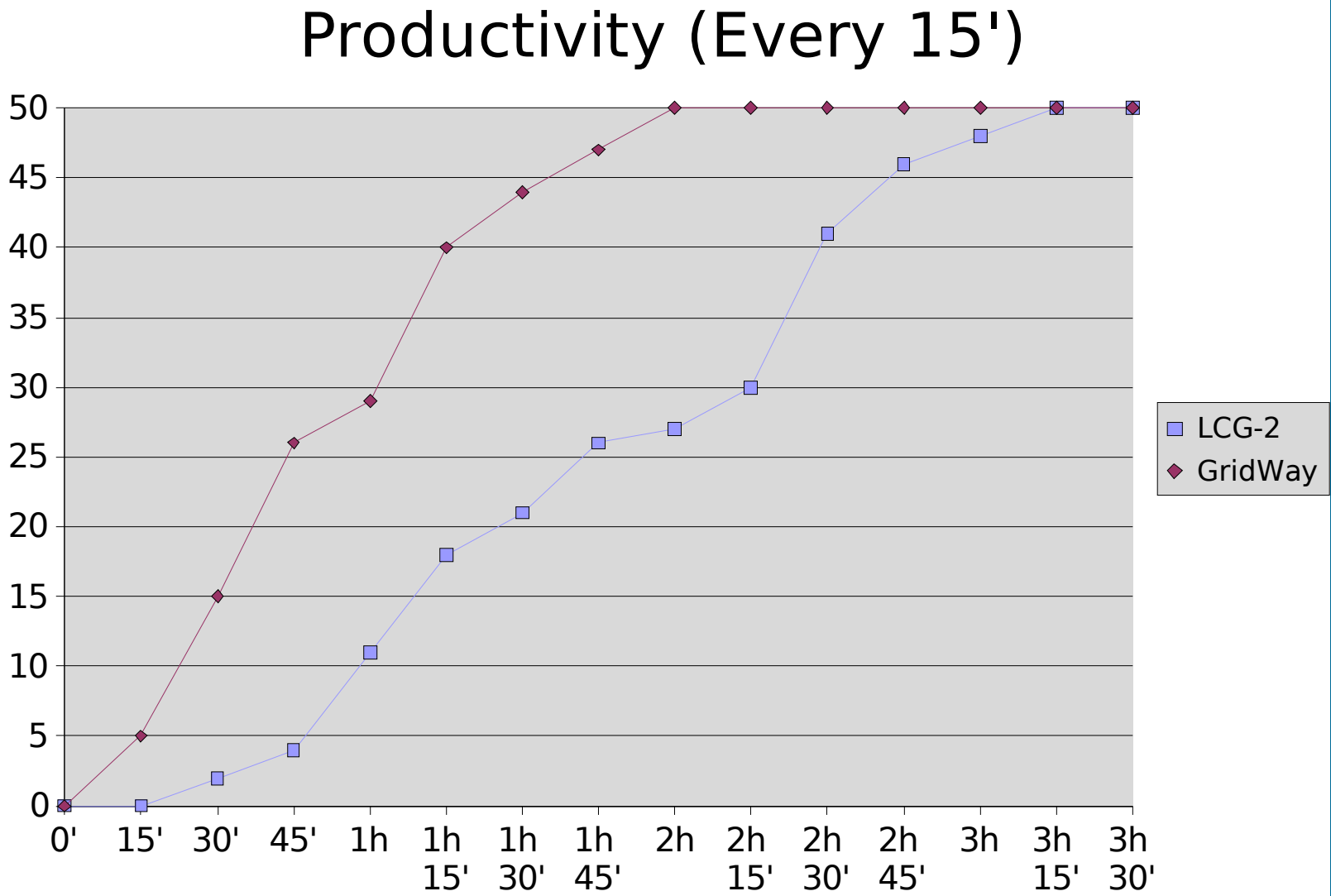



**REMEMBER: With GridWay, only 1 job failed (and was resubmitted)**

# Comparison


Jobs Allocated (Every 15')
22,5
20
17,5
15
12,5
10
7,5
5
2,5
0
0'
15'
30'
45'
1h
1h 15'
1h 30'
1h 45'
2h
2h 15'
2h 30'
2h 45'
3h
3h 15'
3h 30'
LCG-2
GridWay

# Comparison


Productivity (Every 15')
50
45
40
35
30
25
20
15
10
5
0
0'
15'
30'
45'
1h
1h 15'
1h 30'
1h 45'
2h
2h 15'
2h 30'
2h 45'
3h
3h 15'
3h 30'
LCG-2
GridWay

# Conclusions

- GridWay obtains higher productivity
  - Reduces number of nodes and stages
  - Mechanisms not given by LCG-2
    - Opportunistic migration
    - Performance slowdown detection
- API's
  - LCG-2: Relays on specific middleware
  - DRMAA implementation: doesn't
    - GGF standard
    - Job sync, termination and suspension

# “Our two cents”

- Data from Information System should:
  - be updated more frequently
  - represent the “real” situation

# Thank you for your attention!


**Want to give GridWay a try? Download it!**