DIRAC testbed

2nd SuperB Collaboration Meeting @ INFN-LNF

14/12/2011

Bruno Santeramo – INFN-BARI

Outline

- Motivation
- A brief history of DIRAC
- DIRAC community
- Features of DIRAC
- Strengths and Weakness of DIRAC
- DIRAC testbed @ SuperB
  - Use case: analysis
  - Use case: MC production
- Future plans
- People involved

Motivation

- SuperB has not yet a well defined computing model, so we can evaluate what “market” can offers before to decide
- Why evaluate DIRAC ?
  - It's used by two “similar” experiments: LHCb and Belle II
  - It can manage almost all aspect of an experiment's distributed computing
- What evaluate in DIRAC ?
  - Can manage SuperB sites ? (gLite and OSG)
  - Can satisfy requirements for experiment's use cases ?
  - How hard/easy is to manage a DIRAC installation ?

Brief history of DIRAC

1)Initially developed as a System for MC production @ LHCb using pilot jobs

2)DIRAC became the Workload and Data Management System @ LHCb

3)Separation of core and LHCB specific functionalities → DIRAC can be used by other VOs

4)DIRAC v5 → many VOs adopt DIRAC to interact with grid

5)DIRAC v6

- released in Nov 2011
- Many bugfixes, many new functionalities
- Moving towards the creation of a community of DIRAC users

DIRAC community

Subject	Type	status
Belle II	collaboration/experiment	production
BEPC	collaboration/experiment	
BES	collaboration/experiment	
BIOMED @ CREATIS, Lyon	topical community	
CC/Lyon	regional community	
CESGA	regional community	
CTA	collaboration/experiment	
GISELA	regional community	
GLAST	collaboration/experiment	
ILC	collaboration/experiment	
IOIT/Hanoi	regional community	
LHCb	**collaboration/experiment**	**production**
SuperB	collaboration/experiment	testing

Features of DIRAC

- A single point to control/manage DIRAC
- Written in pure Python
- Native use of Pilot job
  - Jobs can works in filling mode
- Complete management of Data and Workload
  - Production activities (simulation, reconstruction, re-processing, etc..)
  - User data analysis
  - Data storage, replication, movement, catalogue, integrity check
  - Software distribution
  - Monitoring and statistics
    - Jobs and pilot efficiency, etc...
    - Data transfer, storage usage, etc...
- Easily extensible via Agents and Services
- Interaction with almost all grid flavour (gLite, OSG, ARC)
- Interaction with cloud (via VM module)
- Interaction with Ganga

Weakness of DIRAC

- Additional software layer over grid services
  - Need a knowledge of DIRAC in addition to "classic" grid
- Lack of detailed documentation, but
  - (slowly) documentation is growing
  - good interaction with DIRAC developers to solve problems
- Need a fine setting of pilot parameters
- Pilot must finish before to retrieve its StdOut and StdErr

Strengths of DIRAC

- VO centric (VO manager can easily enable/disable sites, implement priority policies, get detailed info on resources performance)
- Components can be replicated on several servers
- DIRAC realizes the PULL scheduling paradigm and can be regarded as a very large batch system providing
  - Accounting
  - Priority
  - Fairshare
- Strongly modular
- Useful set of API
- Ganga can submit jobs to DIRAC via DIRAC API
- Transparent usage of OSG and gLite sites
- Ready for Grid. Cloud and local usage

DIRAC testbed @ SuperB

- 1 production server installed @ CNAF (DIRAC v6r0)
  https://bbrbuild01.cr.cnaf.infn.it:8443/DIRAC/
- 1 test server installed @ INFN-BARI
- Many SuperB grid sites configured
  - 23 gLite sites and 1 OSG site
  - 24 sites, 50 CEs and 19 SEs
  - Testing DFC (DIRAC File Catalog)

- Tested use cases
  - Analysis
  - MC production

Analysis - 1

Executable used:

- runPhrReduce (thanks to Elisa Manoni)

Input files:

- 1260 files (~190GB in total)

Sandbox:

- both input and output sandbox registered on Dirac File Catalog

Output files:

- 3 files for each job (1 root + 2 txt)

Grid sites used:

- INFN-T1, INFN-PISA

Submitted jobs:

- 126 jobs
- each job will process 10 input files

Analysis - 2

- It is quite inefficient to deal with very small input files:
  - so we packed 10 input files together and stored in catalog (each file ~ 1,6 GB)
  - each job processed only one “packed” file
  - The name of each input file contains a string that is “parametric” (e.g. aa, ab, ac...)
- Each file is replicated using DFC (Dirac File Catalog) in both sites
- Each job can be scheduled on both sites
- Output files automatically stored in a directory in DFC specific for each job
- The executable is transferred as InputSandbox while we exploited the SuperB software installation already available in each site

Analysis - Results

The analysis is carried out using a script that:

- Untar input file
- Set up the needed environment variables
- Run the real executable with needed parameters
- Tar the output files

No need to interact with grid services

126 jobs correctly distributed between INFN-T1 (~60%) and INFN-PISA (~40%)

No jobs failures from the user point of view

Analysis - ToDo

- Automatize the “input package builder”
  - Packing small input files
- Automatically writing the JDL (Dirac provides a useful API set for this task)
- Automatically writing the user script
- Retrieving all the output and merging together

- Typically those operations could be covered by Ganga or similar tools
  - We need to explore this option before writing new code

MC production - 1

parameters used:

- production: 2010_July_test
- analysis: BtoKNuNu
- generator: B+B-_K+nunu
- geometry: DG_4
- background: MixSuperbBkg
- events: 10,000

10 sites, 100 jobs per site → 1000 jobs submitted

each jobs runs ~2-3 hours

MC production - 2

- Input data:
  - 5 files (in total ~3GB) uploaded in Dirac File Catalog (DFC) and replicated on several Storage Elements
- Stageout
  - Output data uploaded @ CNAF and registered in DFC
  - For each job, one sub-folder for output data

MC production - Results

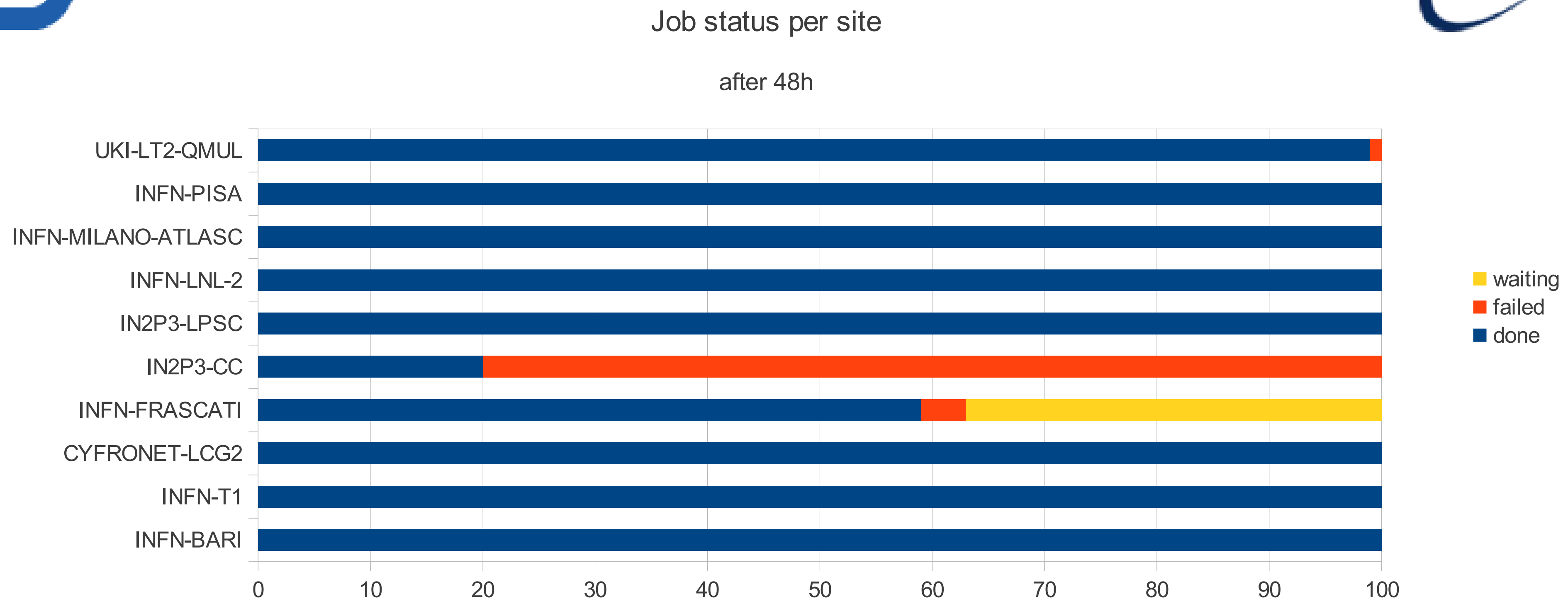
Observation period: 48h

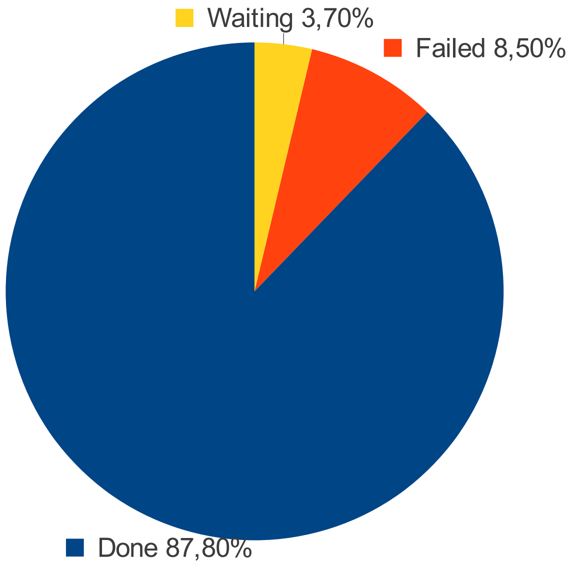
up to 753 jobs simultaneously running

80 jobs failed at IN2P3-CC because submitted to short queue (MaxCPUTime = 6)

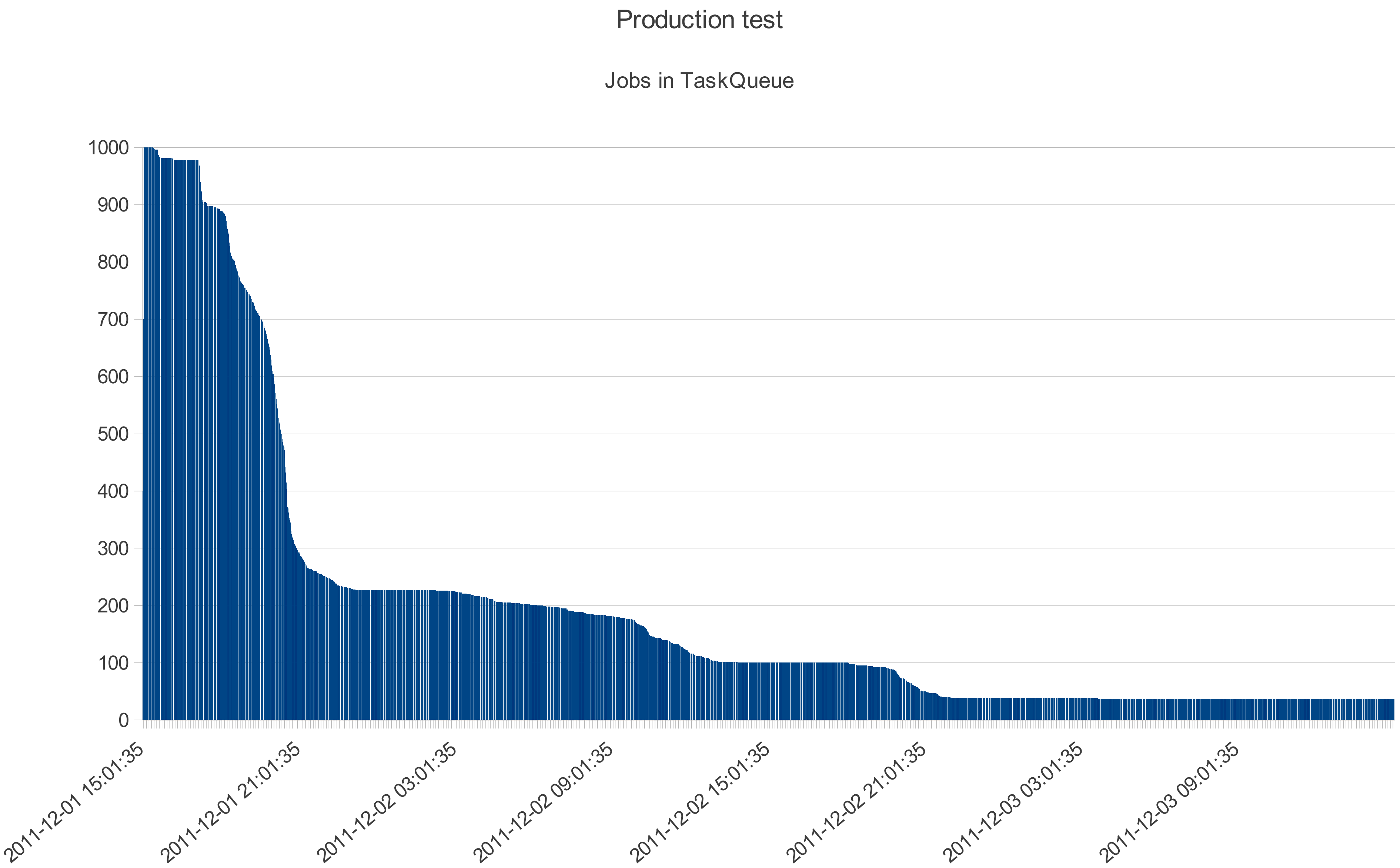
Results similar to current production system used (WebUI+smarty)

MC production - Results





MC production - Results



Future plans

- Fine tuning of pilot's parameters in order to increase responsiveness
- Stress tests (more jobs submitted, more sites involved)
- Increase performance both in analysis and MC production use case
- Test DIRAC data management advanced features:
  - Automatic replication
  - Integrity check
  - Data recovery

People Involved

People working on DIRAC test

- Giacinto Donvito – INFN-BARI
- Bruno Santeramo – INFN-BARI
- Armando Fella – INFN-PISA

Thanks to:

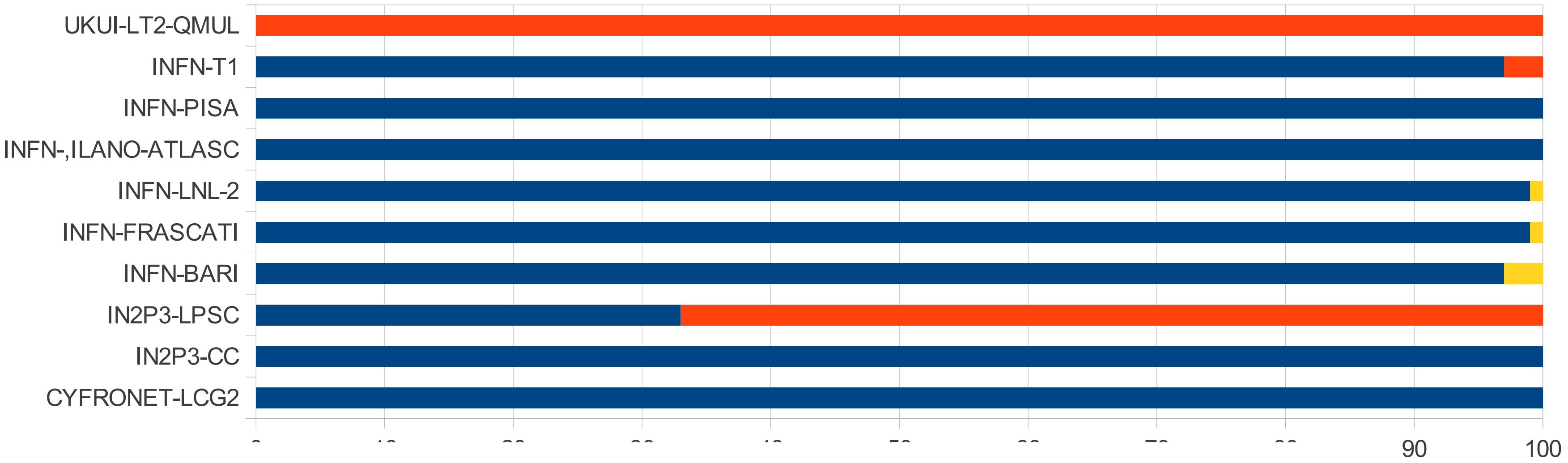
- Matteo Manzali – INFN-FERRARA
- Andrei Tsaregorodtsev – IN2P3-CC
- Ricardo Graciani – Universitat de Barcelona
- the Distributed Computing Group

Thank You

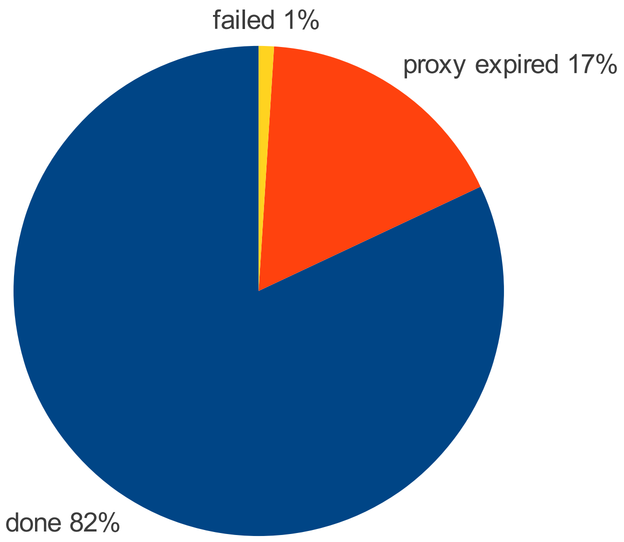
BACKUP SLIDES

MC production - Results

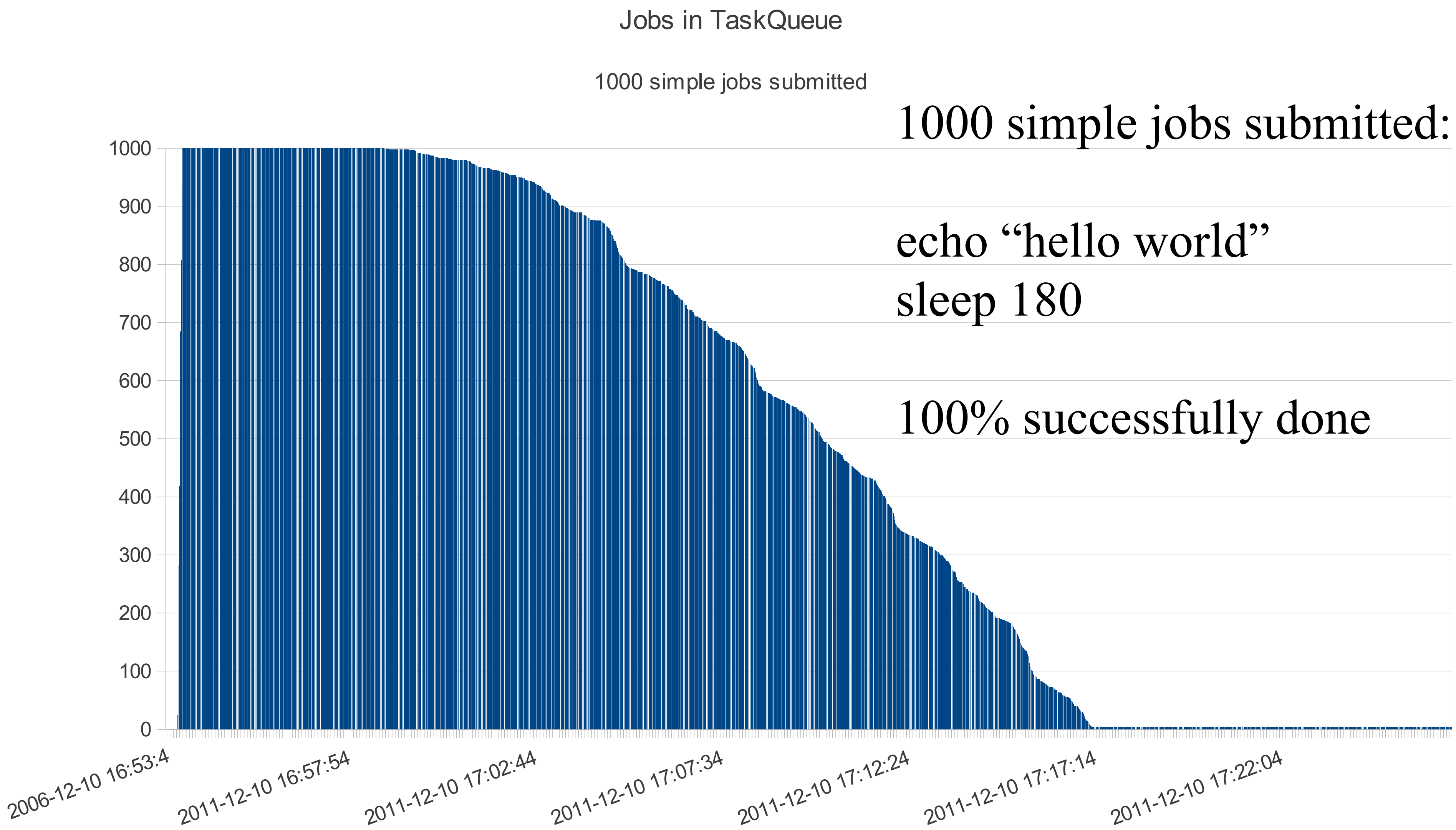
WebUI + smarty



WebUI + smarty



Jobs in DIRAC



1000 simple jobs submitted:

echo “hello world”
sleep 180

100% successfully done

Sites used in MC test

CE	SE
INFN-LNL-2	PADOVA-INFN
CYFRONET-LCG2	CYFRONET-LCG2
IN2P3-CC	IN2P3-CC
INFN-BARI	BARI-INFN
INFN-MILANO-ATLASC	INFN-MILANO-ATLASC
INFN-PISA	PISA-INFN
INFN-T1	CNAF
UKI-LT2-QMUL	
INFN-FRASCATI	
IN2P3-LPSC	

MC production - Results

All output data stored at CNAF and uploaded into DFC

- LFN:/superbvo.org/test_dir/output_testbed/%s/framework.root
- LFN:/superbvo.org/test_dir/output_testbed/%s/BtoKNuNu.root
- LFN:/superbvo.org/test_dir/output_testbed/%s/SemiLepKplusNuNu.root

%s is a parameter (can be used runnumber)

files to use in test

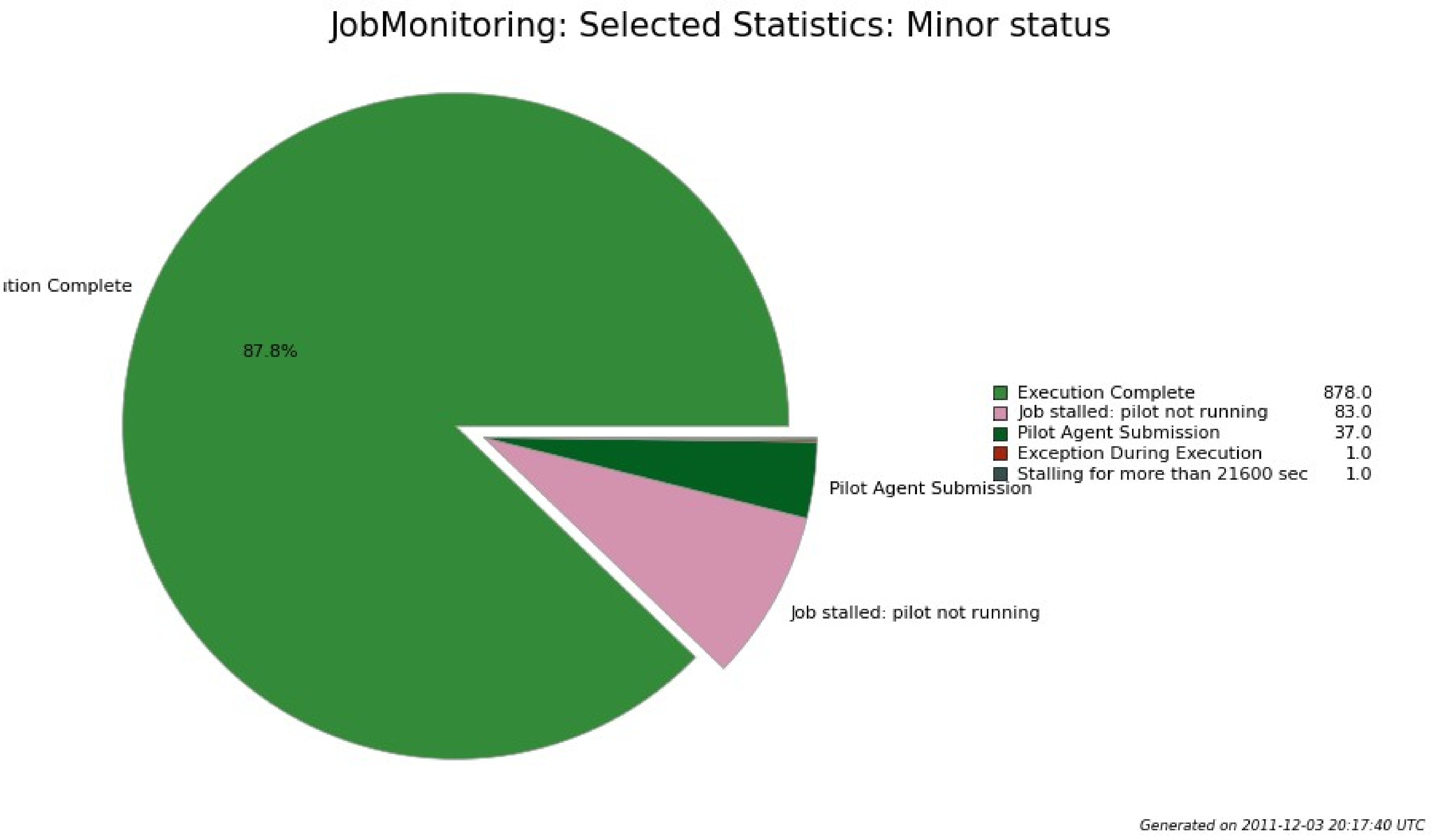
input files:

- lfn:/grid/superbvo.org/production/FastSim/2010_September/input/Bruno_RadBhabha.root
- lfn:/grid/superbvo.org/production/FastSim/2010_September/input/PmcBhabhaBkg.root
- lfn:/grid/superbvo.org/production/FastSim/2010_September/input/PmcBhabhaBkg_DG_4.root
- lfn:/grid/superbvo.org/production/FastSim/2010_September/input/PmcPairBkg.root
- lfn:/grid/superbvo.org/production/FastSim/2010_September/input/PmcPairBkg_DG_4.root

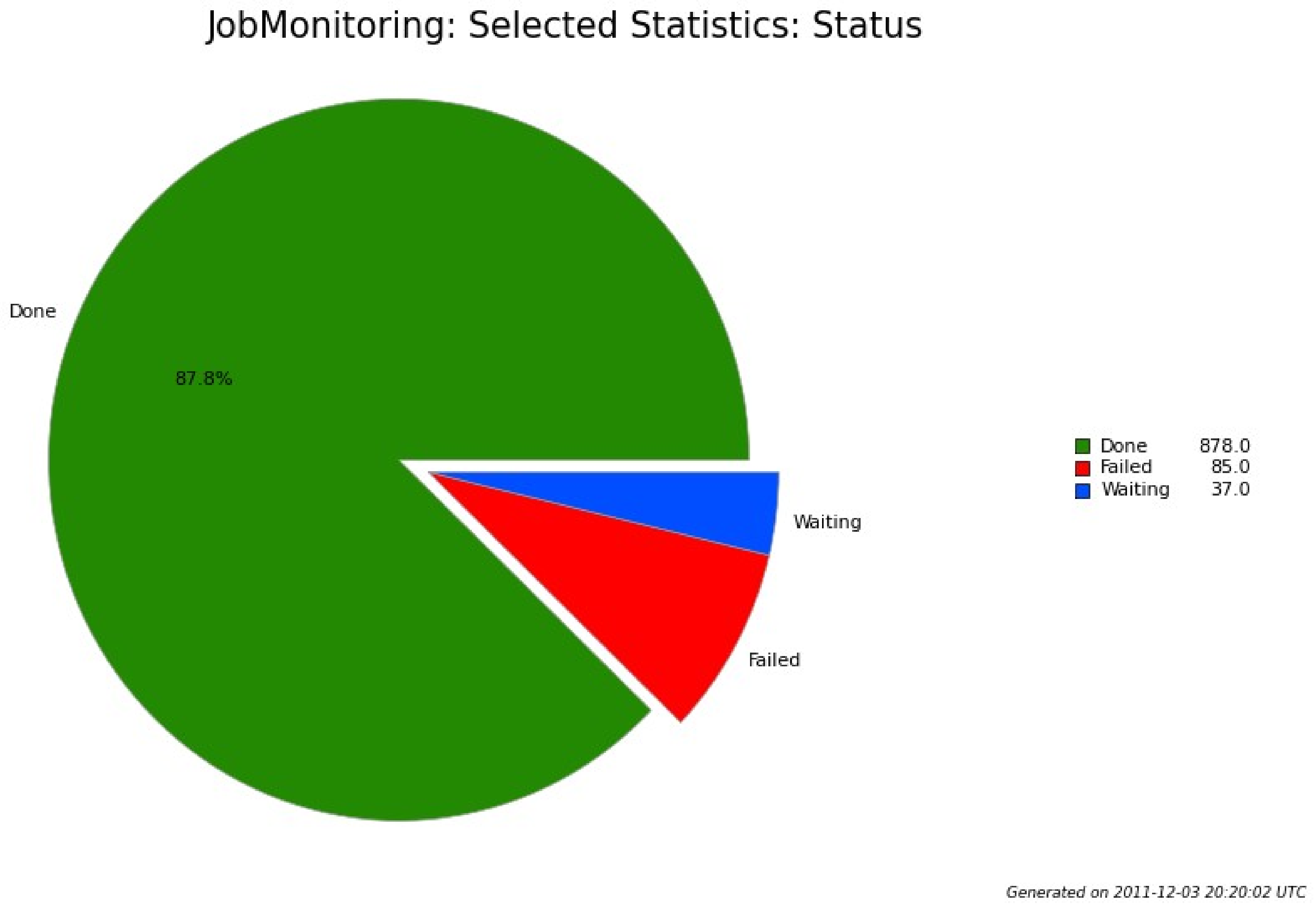
software:

- lfn:/grid/superbvo.org/production/FastSim/2010_September/test_release/V0.2.5_SL-5.3_x86_64_rev311.tjz

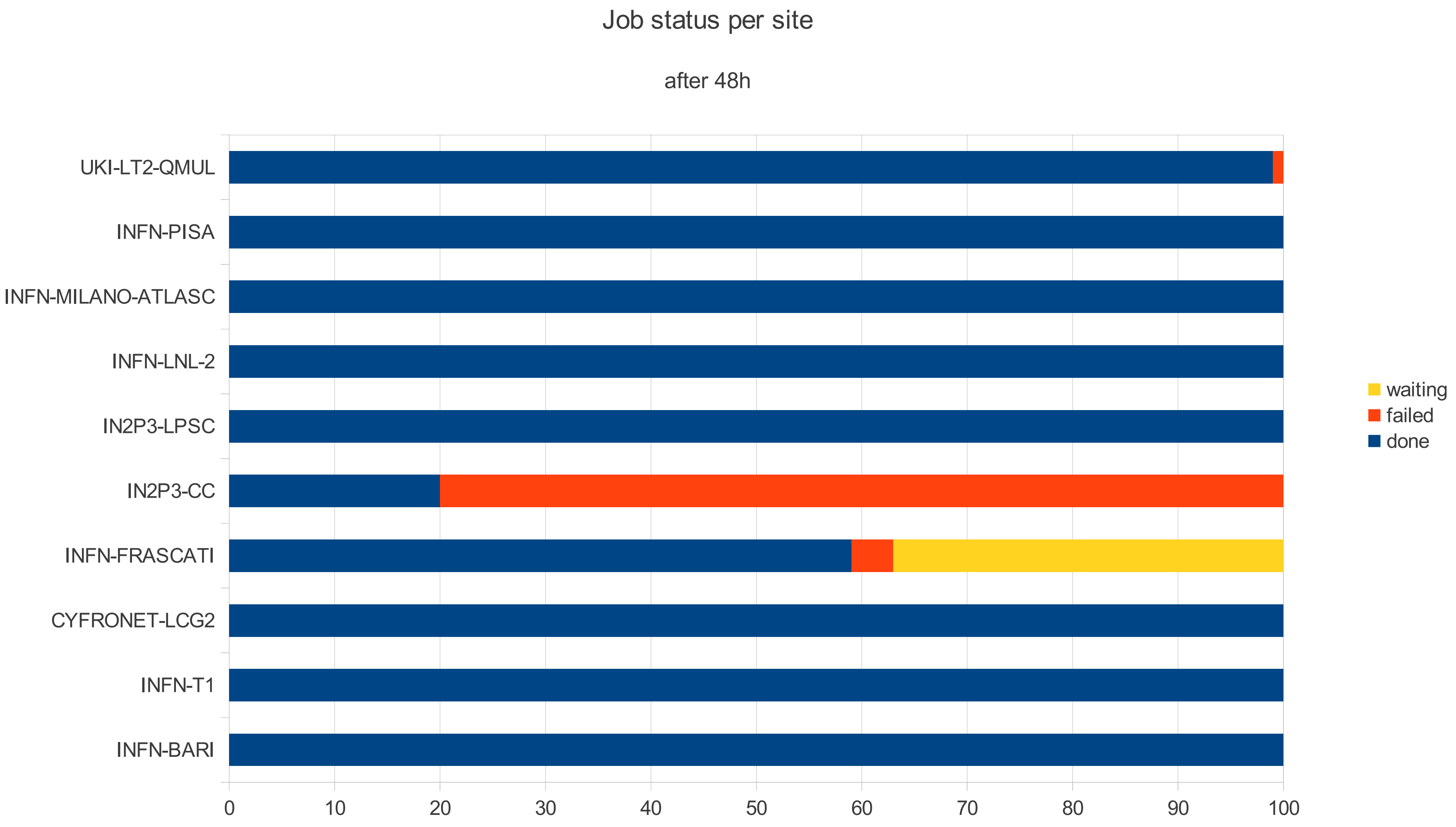
MC production - Results



MC production - Results



MC production - Results



DIRAC & cloud @ Belle II

Cloud test:

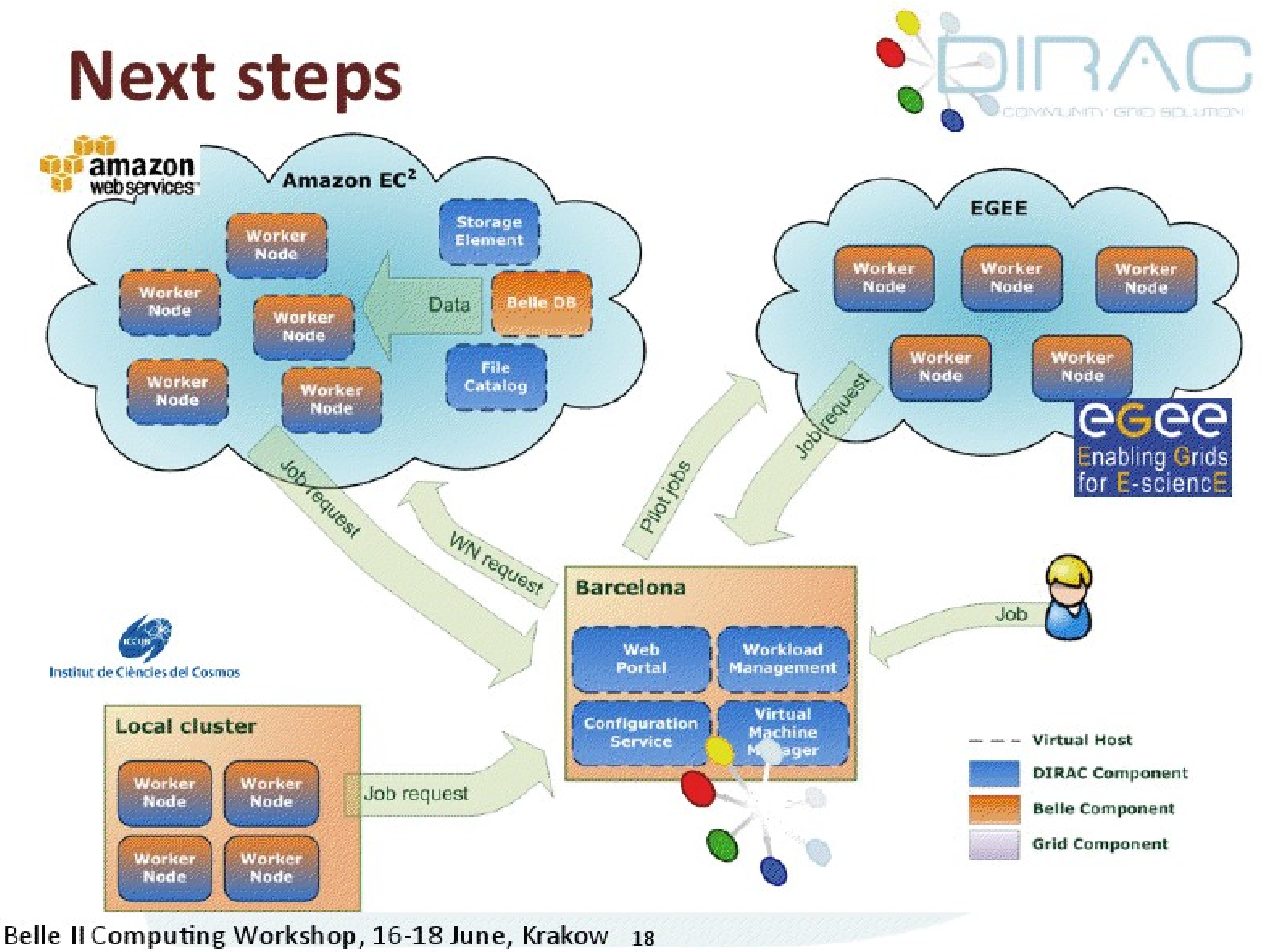
- Poduction ready
  - 5% of Belle production in 10 days
  - 120M evt (~2,7 TB)
  - 2250 CPU days used
- Proven stability and scalability:
  - 2000 CPUs peak achieved in < 4 hours
  - > 90% efficiency in CPU usage
- First cost estimation:
  - **0,46 USD/10k evt**
- Input data pre-uploaded to Amazon SE VM

DIRAC & cloud @ Belle II

Local + Cloud test:

- Poduction ready
  - 7% of Belle production in 6 days
  - 170M evt (~3,6 TB)
  - 3100 CPU days used
- Proven interoperability:
  - 60% cloud resources / 40% local resources
  - > 95% efficiency in CPU usage
- First cost estimation:
  - **0,20 USD/10k evt**
- Input data downloaded from KEK SE

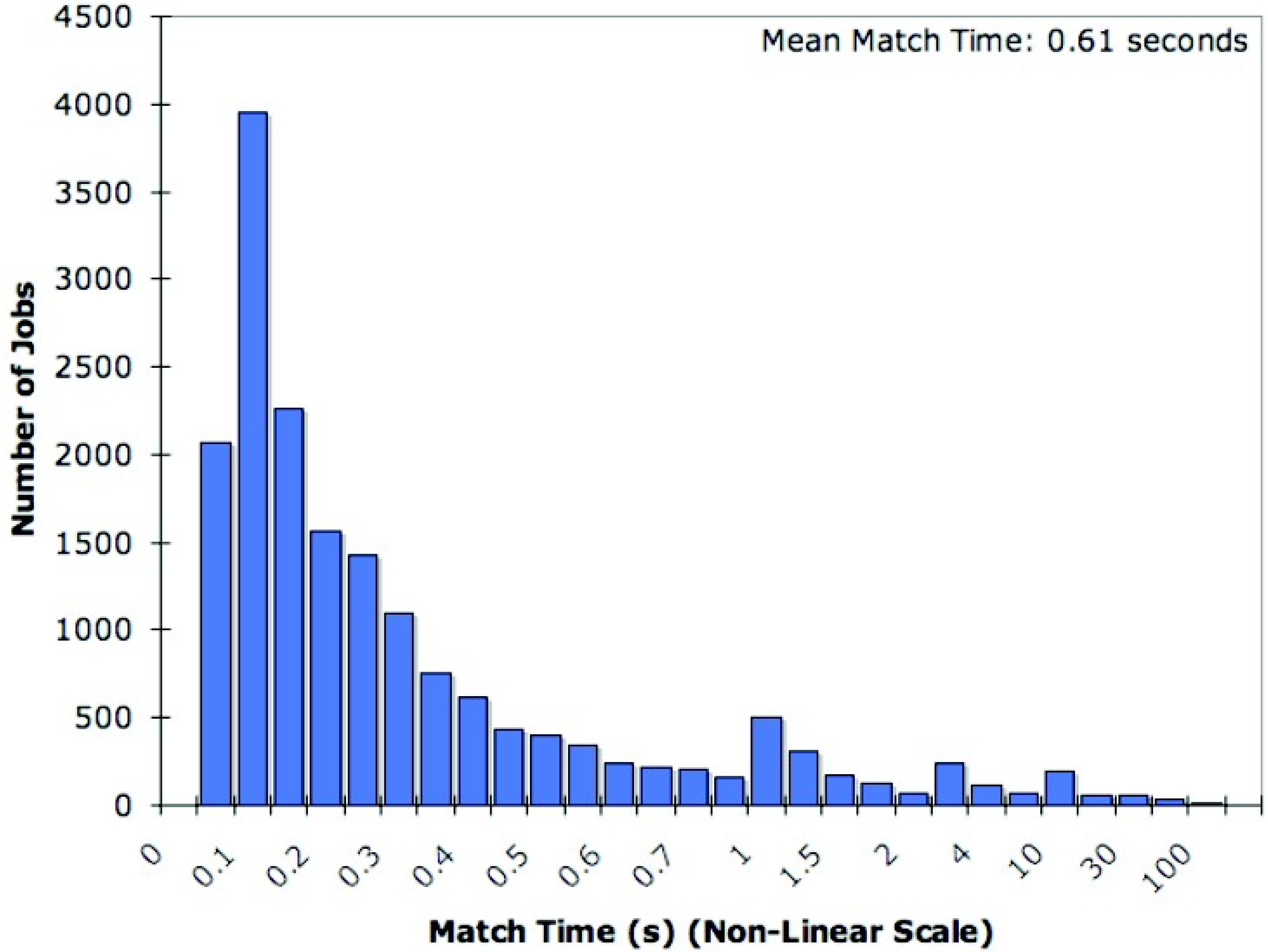
DIRAC & cloud @ Belle II



Next steps

Belle II Computing Workshop, 16-18 June, Krakow 18

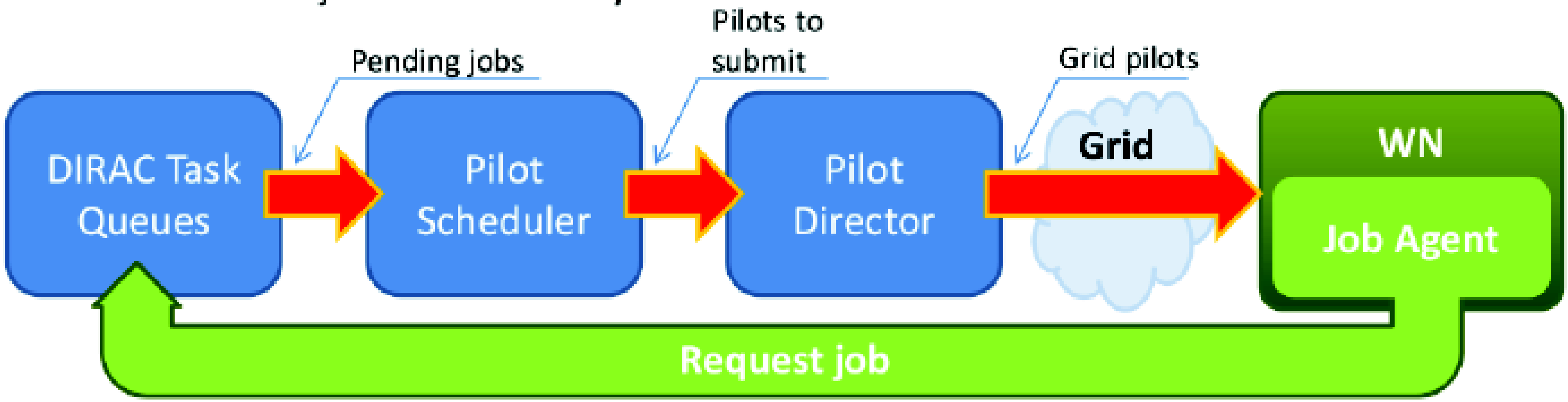
Job's matching @ LHCb



User jobs with many varied requirements present the biggest challenge for the PULL paradigm. DIRAC matcher times for 18K real user jobs submitted between January and August 2007. 92% of jobs are scheduled in under 1 second.

Pilot's job paradigm

- Standard DIRAC job submission cycle


Pilots to submit
Pending jobs
Grid pilots
DIRAC Task Queues
Pilot Scheduler
Pilot Director
Grid
WN
Job Agent
Request job