Belle and the Amazon EC2 cloud

Martin Sevior, University of Melbourne

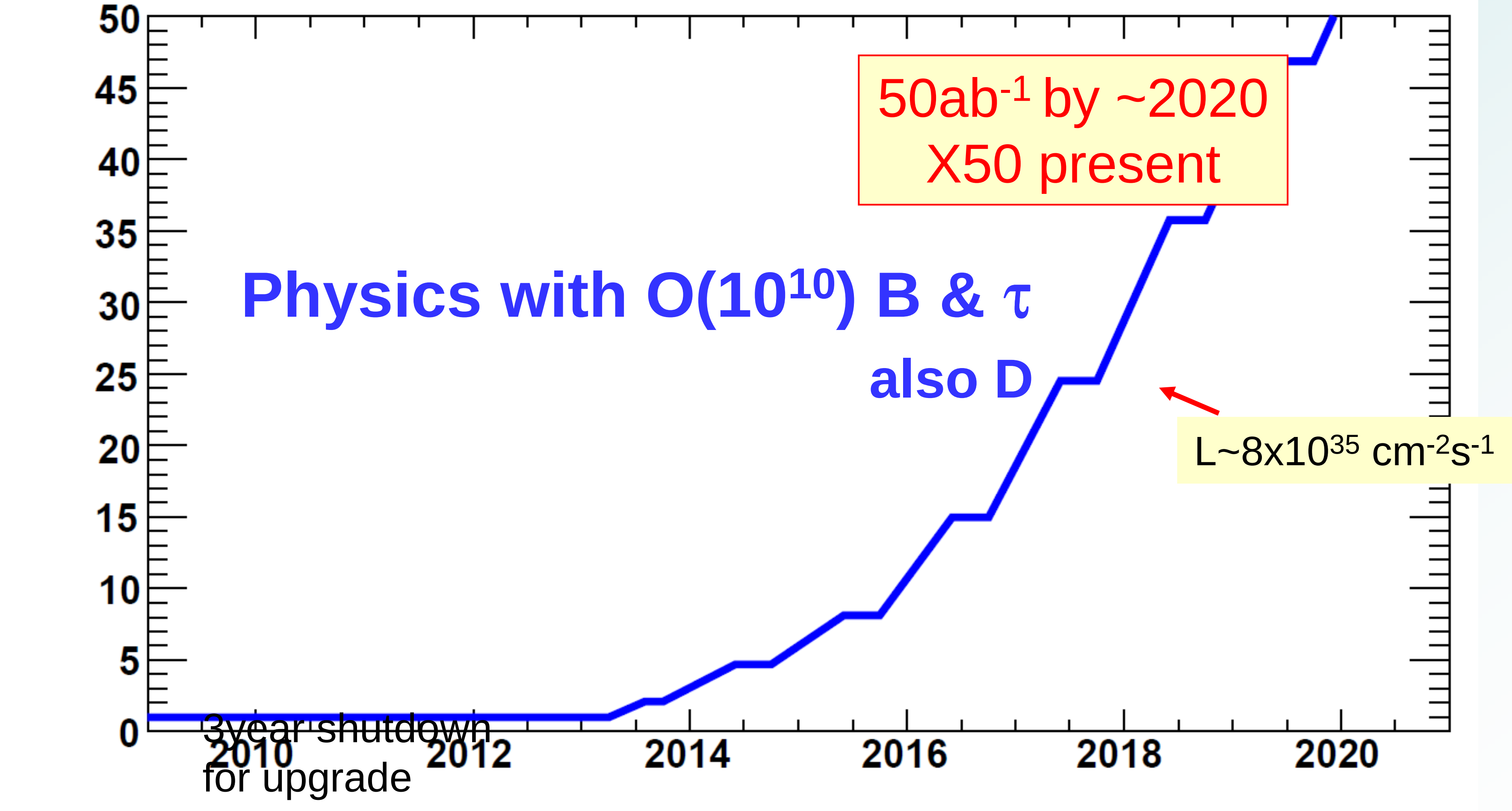
(Tom Fifield University of Melbourne,
R. Graciani Diaz, A. Casajus Ramo, A. Carmona Agüero
University of Barcelona)

DPHEP KEK, July 2010

Outline

- Computing Requirements of Belle II
- Value Weighted Output
- Commercial Cloud Computing – EC2
- Belle MC production
- Implementation of Belle MC on Amazon EC2 -1
- DIRAC and Belle MC on EC2
- Latest results & costs of EC2

Expected Luminosity at Belle II



Current KEKB Computer System

Data size ~ 1 ab^{-1}

New KEK Computer System has 4000 CPU cores

Storage ~ 5 PetaBytes

Belle II Requirements

Initial rate of $2x10^{35}$ cm^2sec^{-1}=> 4 ab^{-1} /year

Design rate of $8x10^{35}$ cm^2sec^{-1}=> 16 ab^{-1} /year

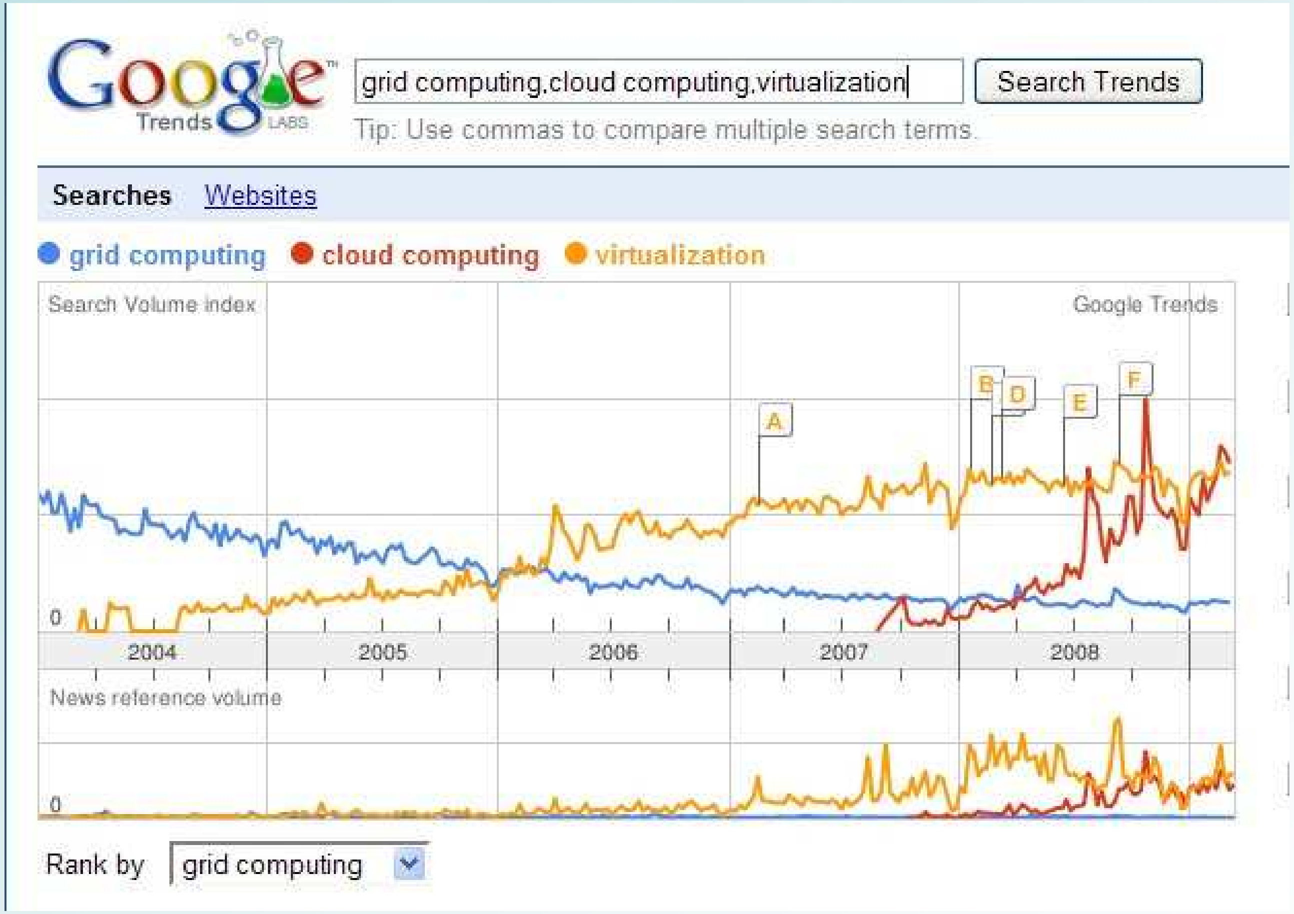
CPU Estimate 10 – 40 times current depending on reprocessing rate

So $4x10^4$ – $1.2x10^5$ CPU cores

Storage 15 PB in 2013, rising to 60 PB/year after 2016

"Cloud Computing"

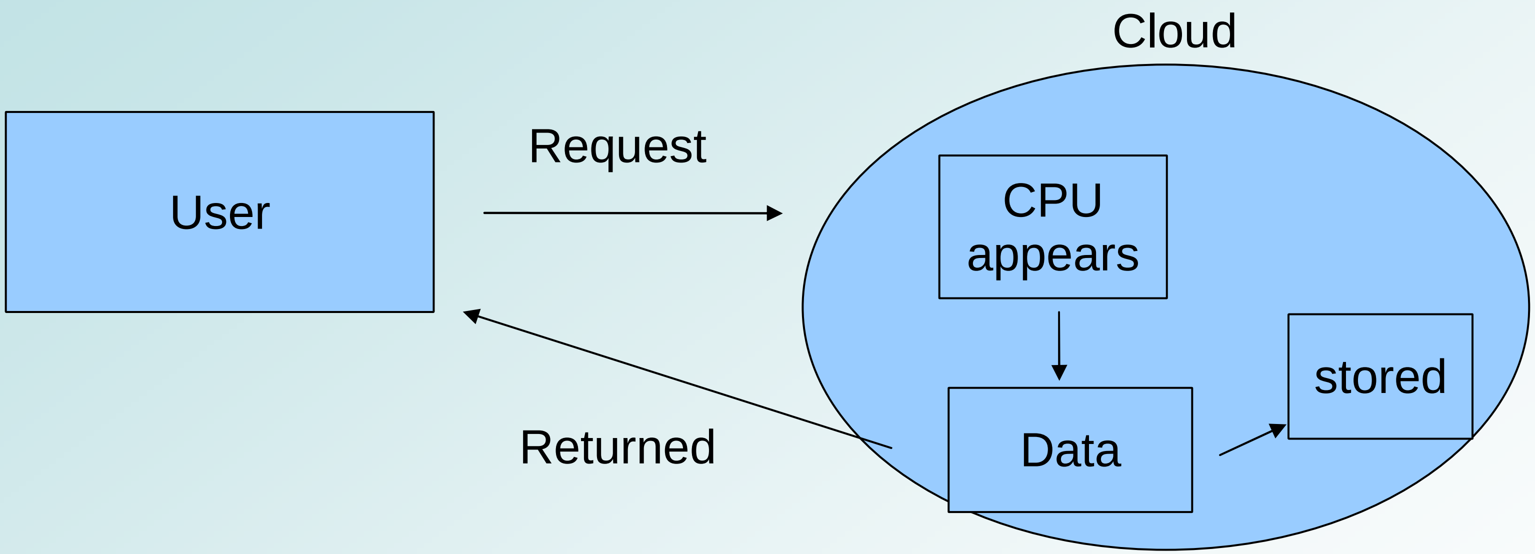
Decided we couldn't ignore Cloud



Can we use Cloud Computing to reduce the TCO of Belle II Computing?

Cloud Computing

Economies of scale
Smaller admin costs.

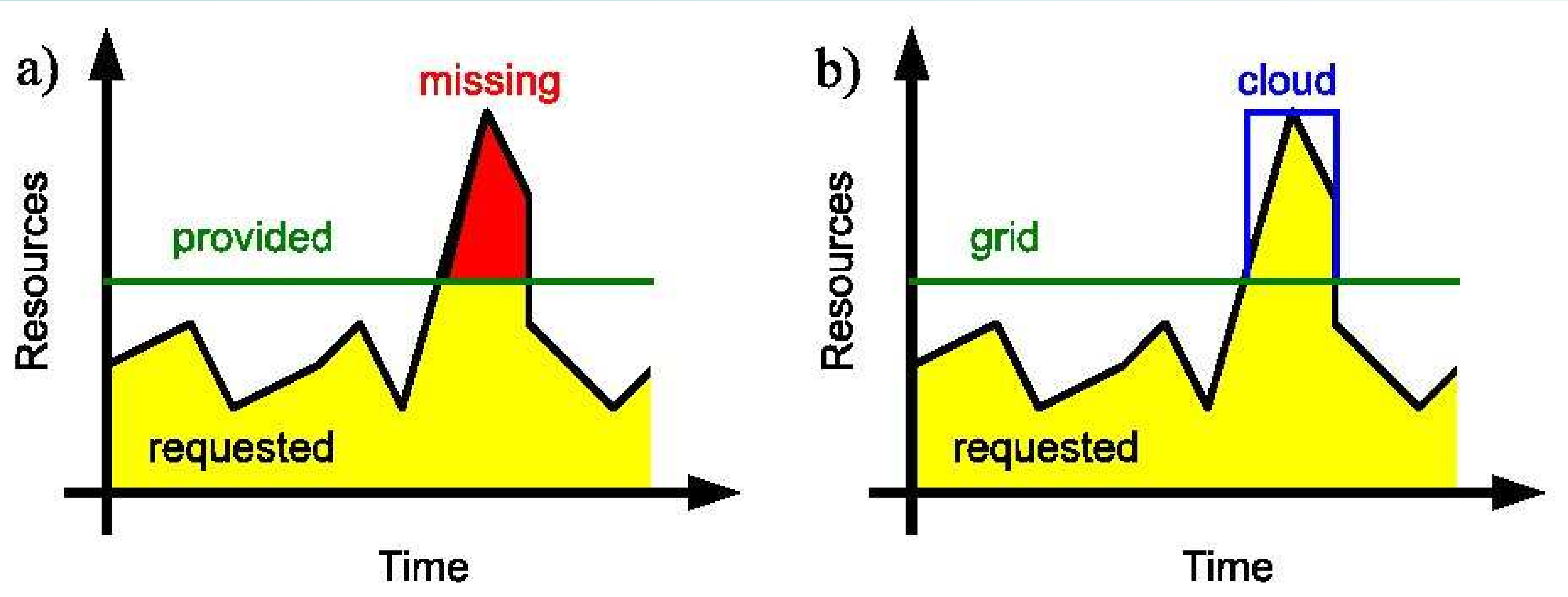


Resources are deployed as needed.

Pay as you go.

MC Production is a large fraction of HEP CPU - seems suited to Cloud

Particularly useful for Peak Demand



Value Weighted Output

- Question: Does the value of a cluster decrease with time?
- Yes! We’ve all seen sad old clusters nobody wants to use.
- How do we quantify how the value of a CPU decreases?
- Moores’ Law? “Computing Power Doubles in 1.5 years”

Moores Law: $P = 2^{t/1.5}$ P = CPU Power, t time in years

$\Rightarrow P = e^{\lambda t}$ $\lambda = 0.462\ \text{years}^{-1}$

Suppose a CPU can produce X events per year at purchase:

Conjecture: The Value of that output drops in proportion to Moores’ Law

Define a concept: Value Weighted Output (VWO)

Value Weighted Output, VWO

So for a CPU with an output of X events per year:

$$VWO = Xe^{-\lambda.0} + Xe^{-\lambda.1} + Xe^{-\lambda 2} + Xe^{-\lambda 3} + ...$$

Truncating after 3 years (typical lifespan of a cluster), gives

$$VWO = \sum_{t=0}^{3} Xe^{-\lambda.t} \cong \int_{0}^{3} Xe^{-\lambda t} = 2.05X$$ (Taking t to infinity gives 2.2 X)

On the other hand the support costs are constant or increase with time

Cloud - Purchase CPU power on a yearly basis.
Always get “current” technology

The legacy kit of earlier purchases need not be maintained

Downsides are well known. Not least of which is Vendor lock in.

Amazon Elastic Computing Cloud (EC2)

- Acronyms For EC2
- Amazon Machine Image (AMI)
  - Virtual Machine employed for Computing
  - ($0.1 - $0.68 per hour of use)
- Elastic Block Store (was S3)
  - $1.8 per Gb per Year (2009),
  - Belle 5 PB ~ $10 million/year
  - => factor 10 too expensive for all data
  - Now $1.2 per GB per year (2010)
- Simple Queuing Service (SQS)
  - Used control Monitor jobs on AMI's via polling (pay per poll)
  - Really cheap!

•Chose to investigate EC2 in detail because it appeared the most mature
•Complete access to AMI as root via ssh.
•Large user community
•Lots of Documentation and online Howto's
•Many additional OpenSource tools

Building the AMI's

- AMI's can be anything you want.
- Many prebuilt AMI's available but no Scientific Linux
- Create Virtual Machines and Store them on S3
- Built 4 AMI's
  - An Scientific Linux (SL) 4.6 instance (Public)
  - SL4.6 with Belle Library (Used in initial Tests) (Private)
  - SL5.2 (Public)
  - SL5.2 with Belle Library (Production, Private)

- We used a loopback block device to create our virtual image.
- Standard yum install of SL but with a special version of tar
- Belle Libraries added to the base AMI's via rpm and yum
- Uploaded to S3 and registered

Initial Tests

- Quick tests to check things and first guess at costs (2009)

Instance Type	EC2CU	RAM	ARCH	$/Hour
m1.small	1	1.7	32 bit	0.10
m1.large	4	7.5	64 bit	0.40
m1.xlarge	8	15	64 bit	0.80
c1.medium	5	1.7	32 bit	0.20
c1.xlarge	20	17	64 bit	0.80 (0.68)

Initial Test results 2009

Machine	cost/10^4 events	cost/10^9 events
Small EC2 Instance	$2.065	$206,541.575
Large EC2 Instance	$1.175	$117,504.489
Extra Large EC2 Instance	$1.176	$117,637.111
HighCPU Med EC2 Instance	$1.029	$102,913.583
HighCPU XL EC2 Instance	$0.475	$47,548.933

10^9 events is approximately the MC requirement of a Belle 3-month run

PowerEdge 1950 8-core box (used in Melbourne Tier 2) Cost ~ $4000
10^4 events in 32 minutes ,

Amortization Period	Events Generated	Cost/10^4 events	VWO Cost/10^4 events
8000 hours - 1 Year	$160x10^6$ events	$0.25	$0.12
16000 hours - 2 Year	$320x10^6$ events	$0.13	
24000 hours - 3 Years	$480x10^6$ events	$0.08	

Electricity consumption: 400 W => 3500 KWhr/Yr ~$700/year in Japan
Over 3 years, VWO cost (with additional electricity) is $0.16 per 10^4 events

Full scale test - 1

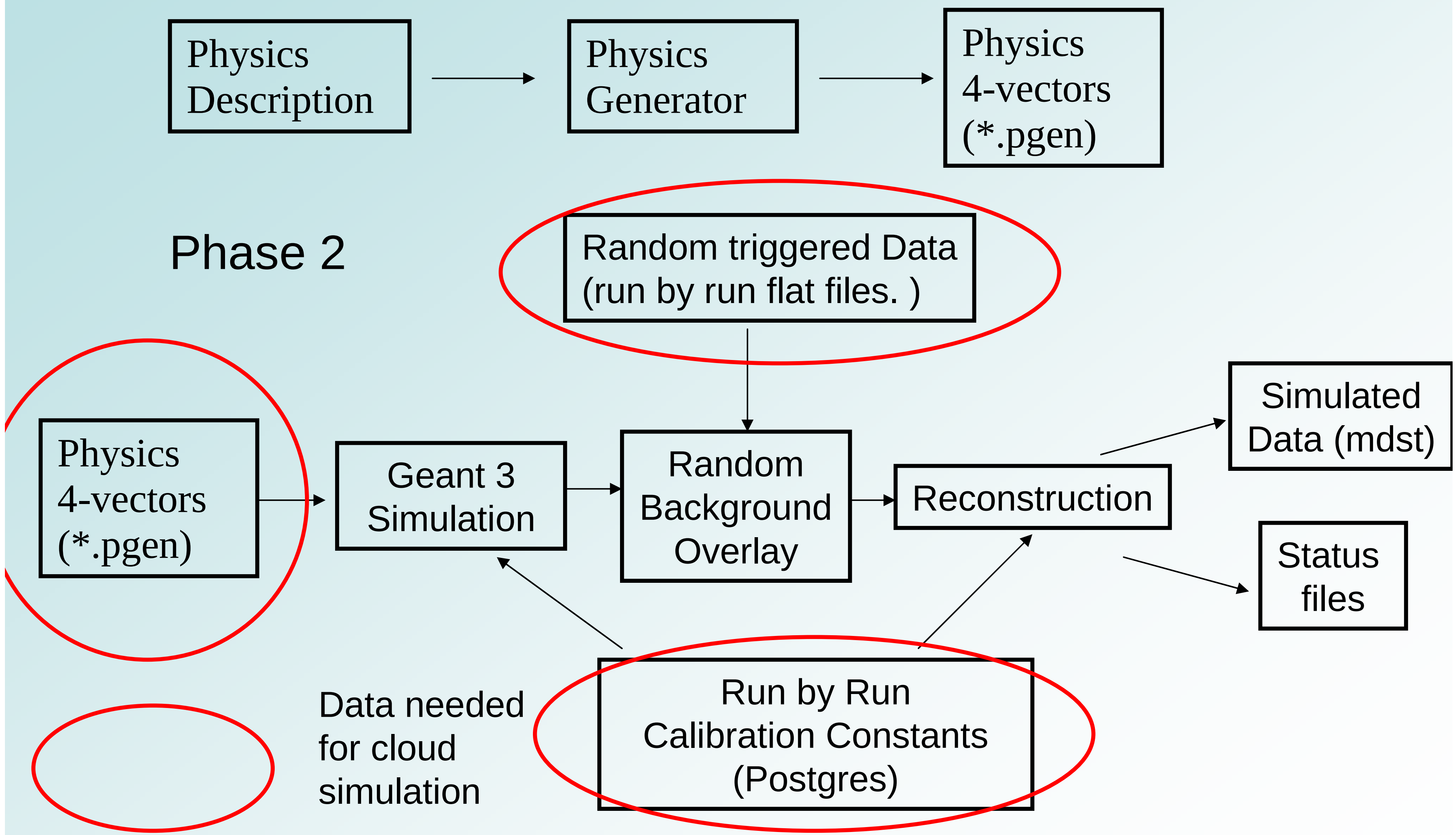
- Initial test was for a series of runs on a single CPU
- Neglected important additional steps as well as startup/shutdown
- Next step was a full scale Belle MC production test.
- Million event Generation to be used for Belle Analysis

Accessing Data on the Cloud

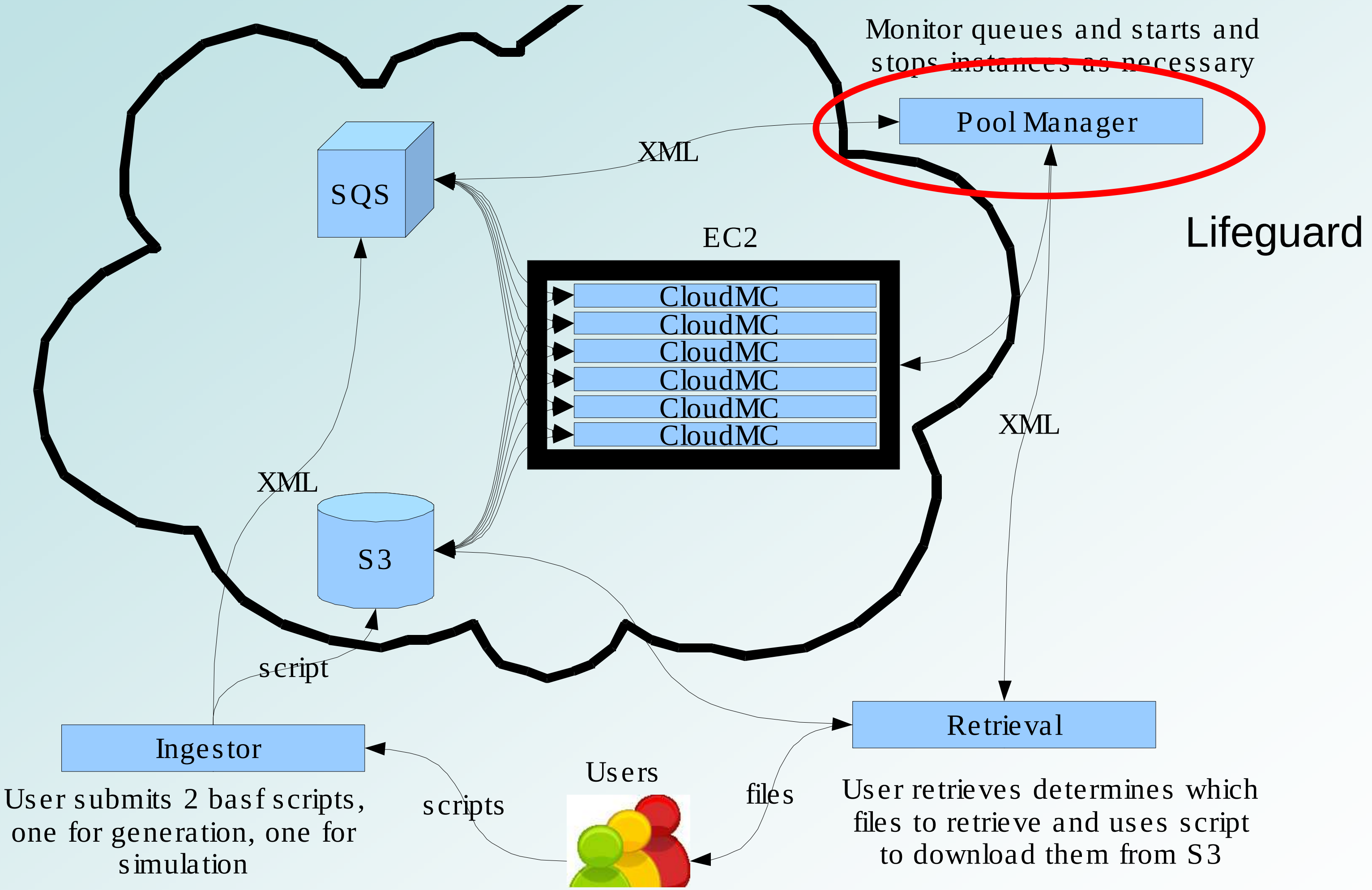
- Full scale Belle MC production requires 3 types of data
- *.pgen files which contain the 4-vectors of the Physics processes
- Random triggered background Data, ("addBG") to be overlayed on the Physics
- Calibration constants for alignment and run conditions
- *.pgen and addBG data were loaded onto S3
- Accessed via a FUSE module and loaded into each AMI instance
- Calibration data was accessed via an ssh tunnel to a postgres server at Melbourne

Full scale Belle MC Production 2009

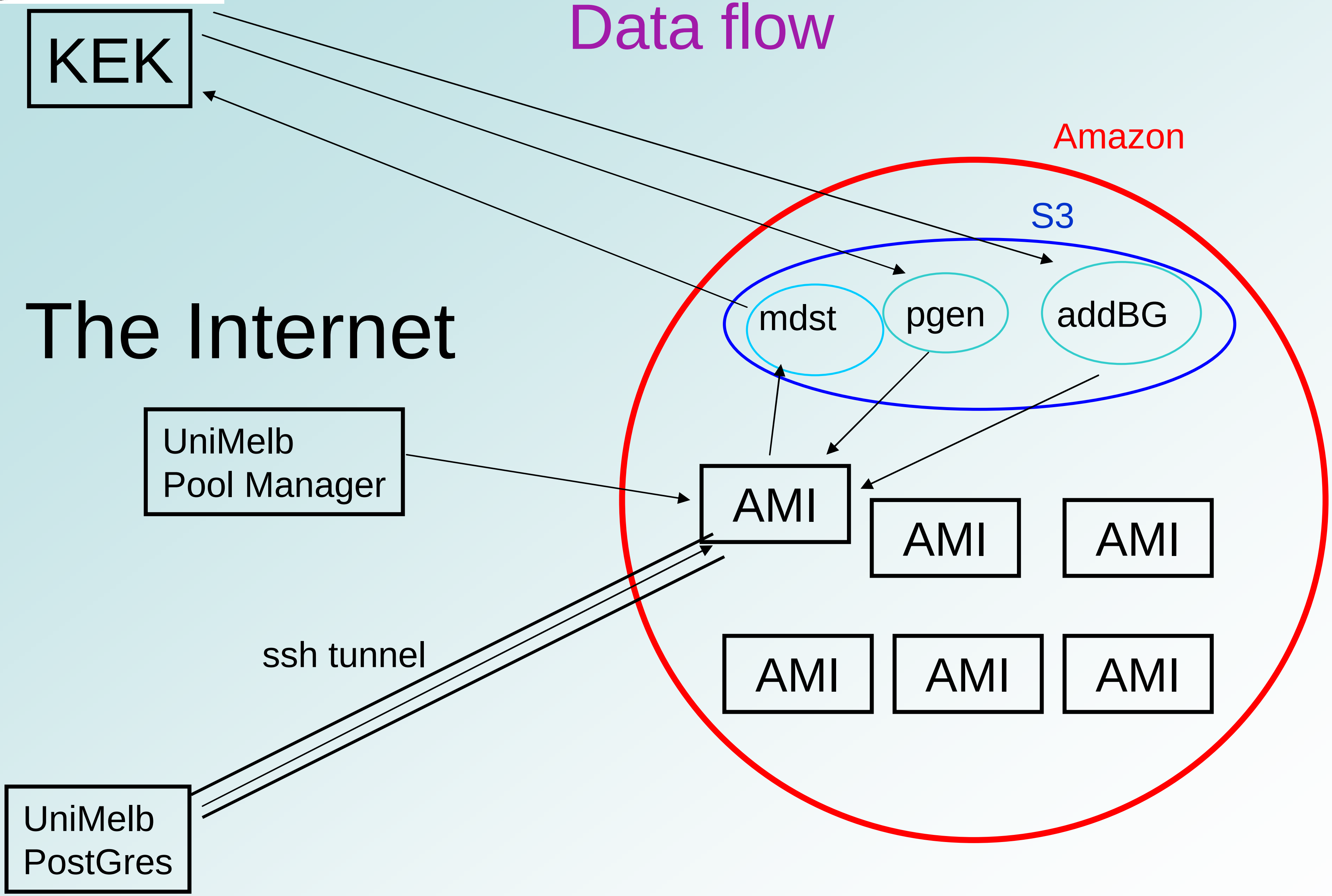
Phase 1



Automating Cloud Production

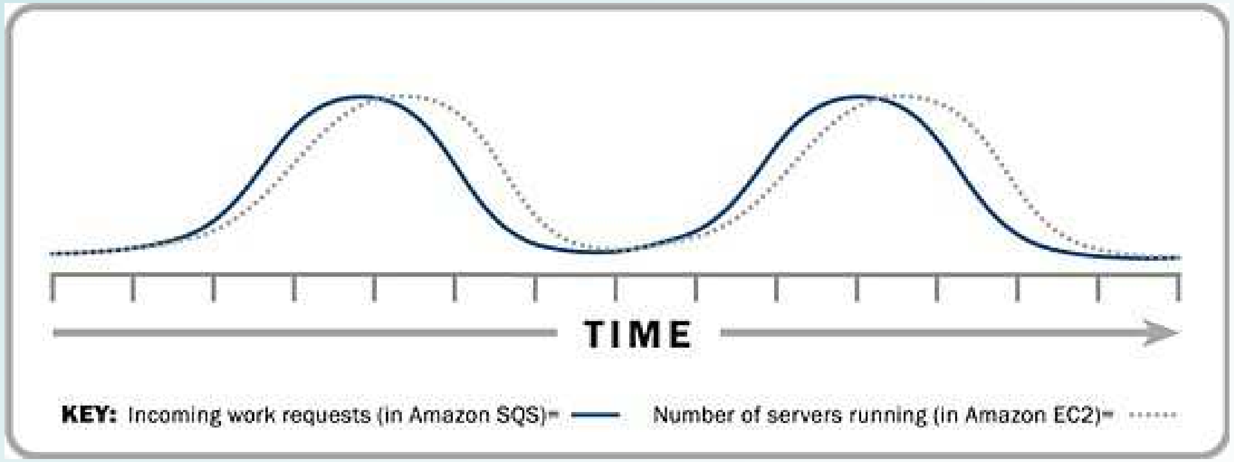


Data flow


KEK
Amazon
S3
mdst
pgen
addBG
The Internet
UniMelb
Pool Manager
AMI
AMI
AMI
AMI
AMI
AMI
ssh tunnel
UniMelb
PostGres

Lifeguard

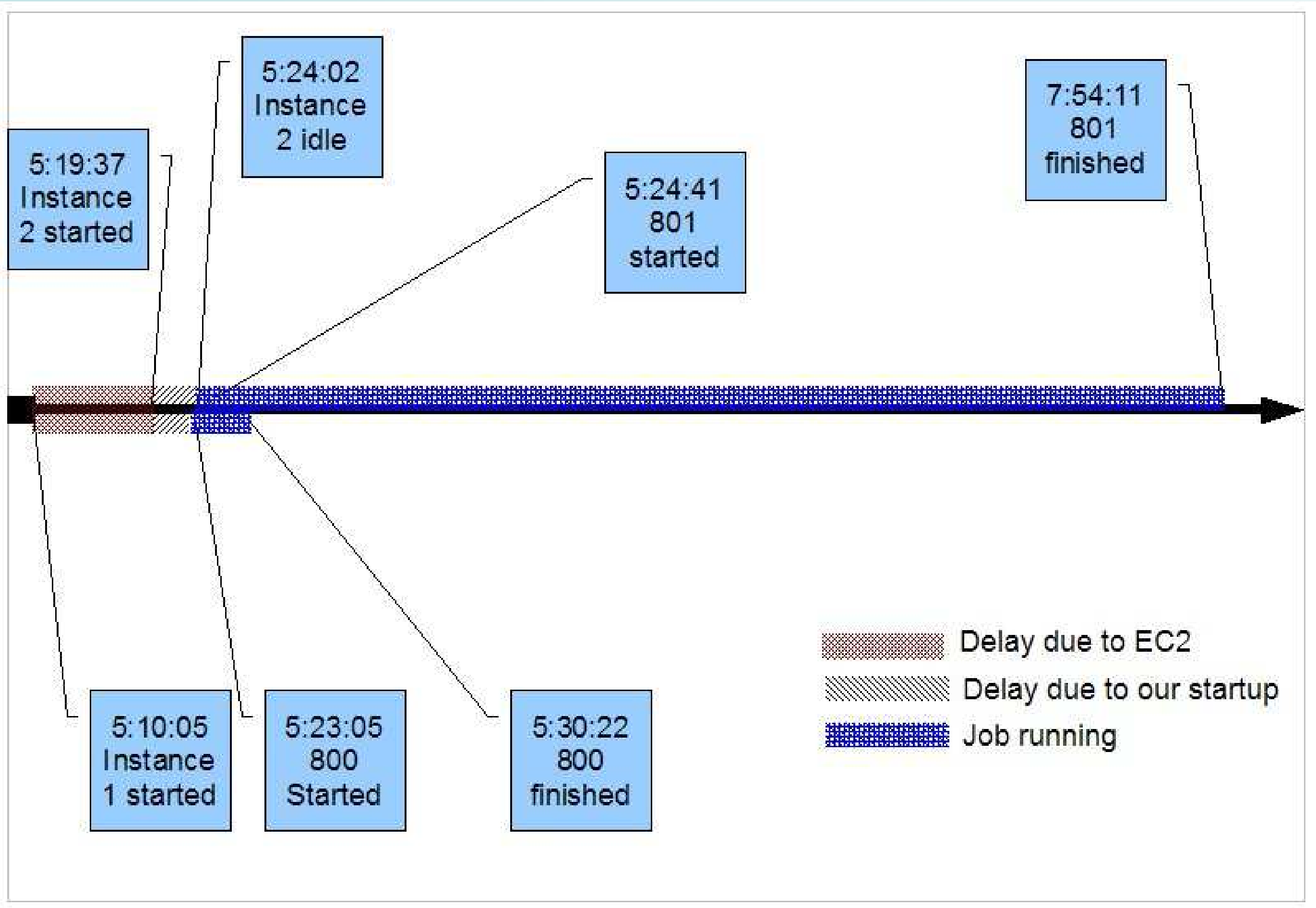
- Employ an Open Source tool called Lifeguard to manage the pool of AMIs.
- Manages the MC production as a Queuing Service
- Constantly monitors the queue
- Starts and stops AMIs as necessary
- Deals with non-responsive AMIs
- Tracks job status



Shutsdown idle AMI's at the end

Dead time and bottlenecks

- Reduce startup time and transfer bottlenecks to minimize costs



Costs Test 1 - 2009

Bottlenecks identified and reduced

- 1.47 Million events generated.
- 16% failure rate (needs more investigation)
- 22 hours on wall clock
- 20 instances of 8 cores (160 cores in total)
- 135 Instances hours cost $USD 108
- 40 GB data files transferred to KEK $6.80
- **Total cost per 10^4 events = $0.78**

Scale Test 2 - 2009

- Run 100 instances – 800 simultaneous cores
- Generated ~ 10 million events
- EC2 scaled well, no problem with 100 AMI's
- Lifeguard flakey
- FUSE-S3 module flakey

- Lifeguard pool manager showed scaling issues
- Investigate DIRAC grid framework for EC2

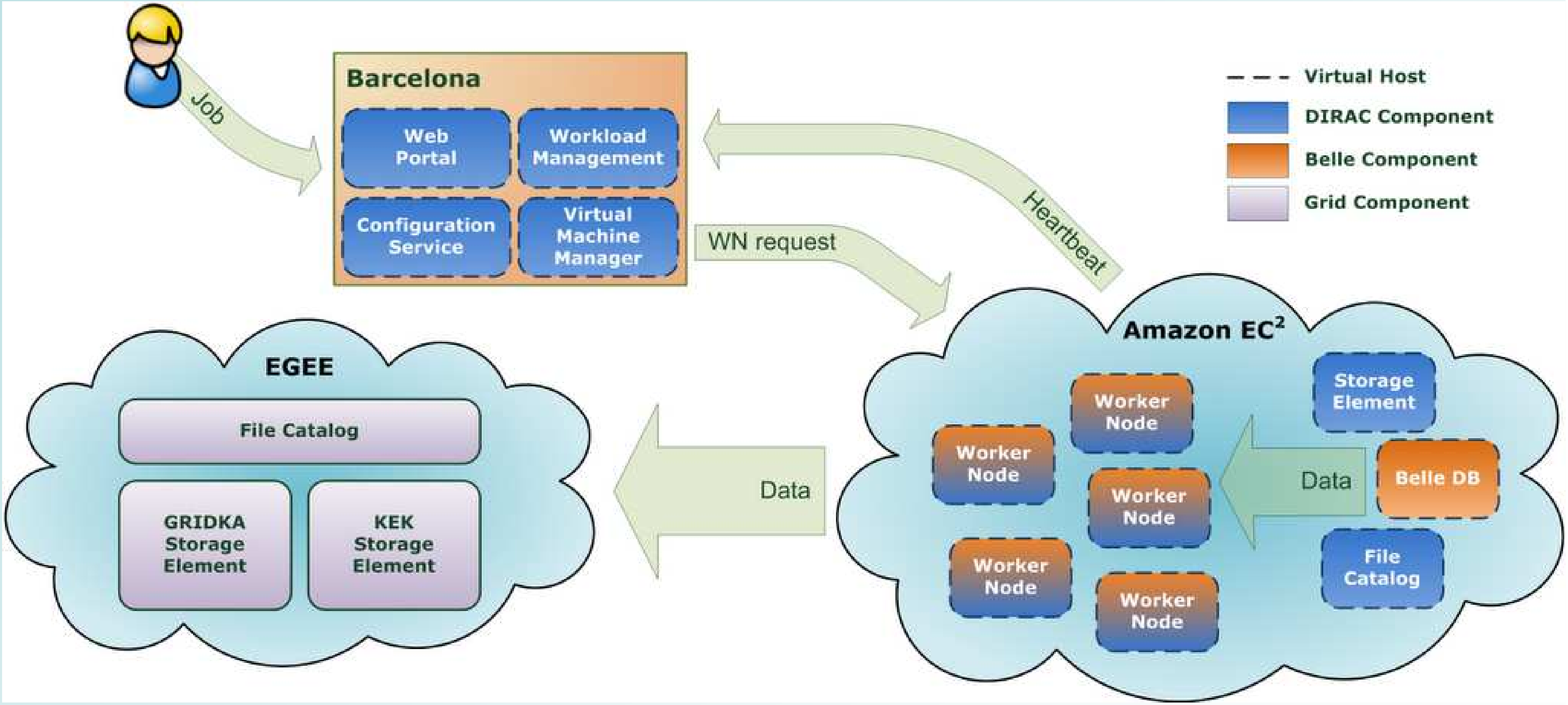
DIRAC

- **VO Centric**
  Gives to the community, the VO, a central role in the relation of its users with their computing resources

.
- **Modularity**:
  To achieve optimal scalability and flexibility, a highly modular design was decided.

- **Pull Scheduling**:
  Implements pull scheduling with late binding of payload to resource to extract optimal performance out of the ever changing underlying resources

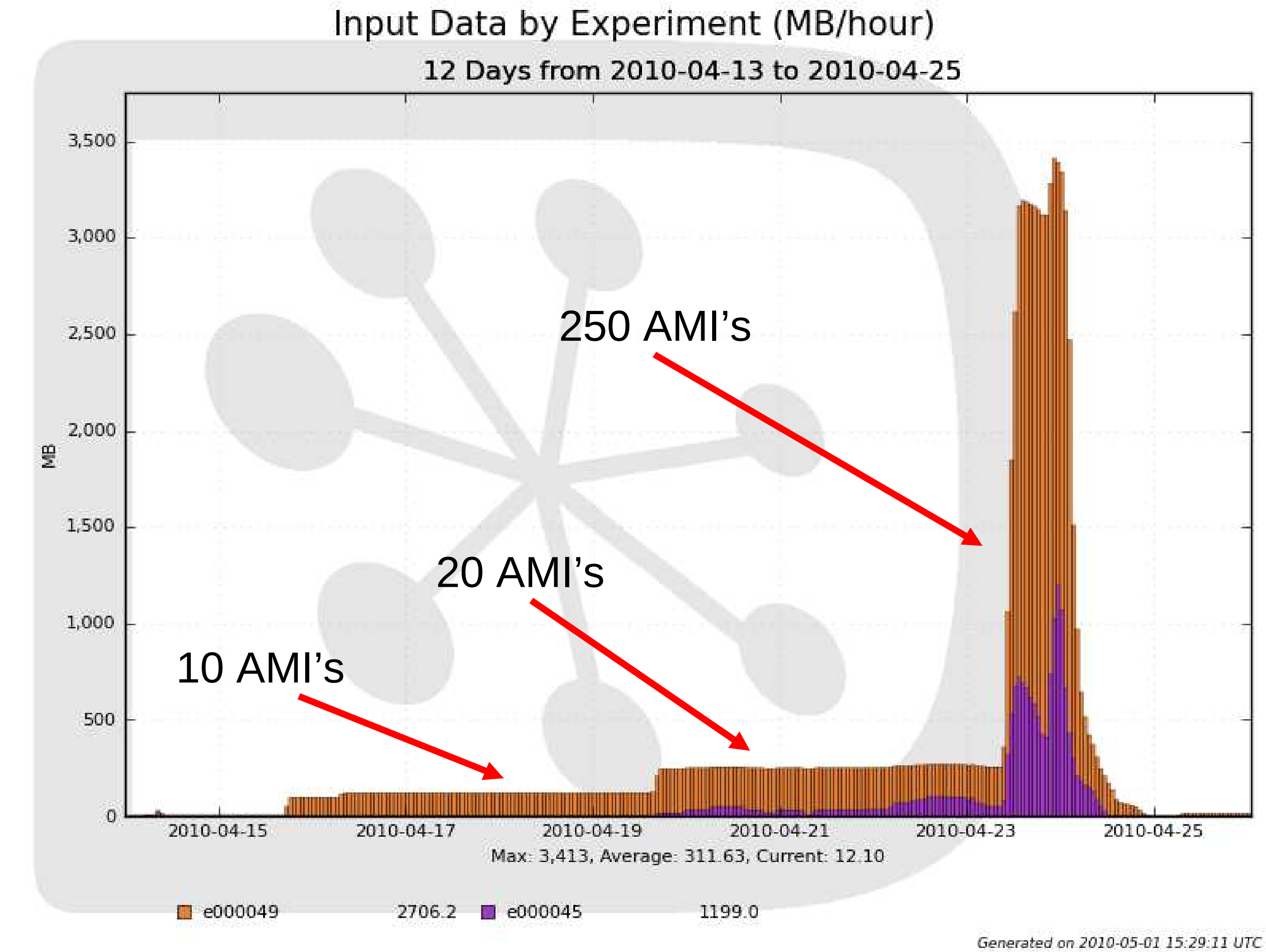
DIRAC Belle - Cloud Solution



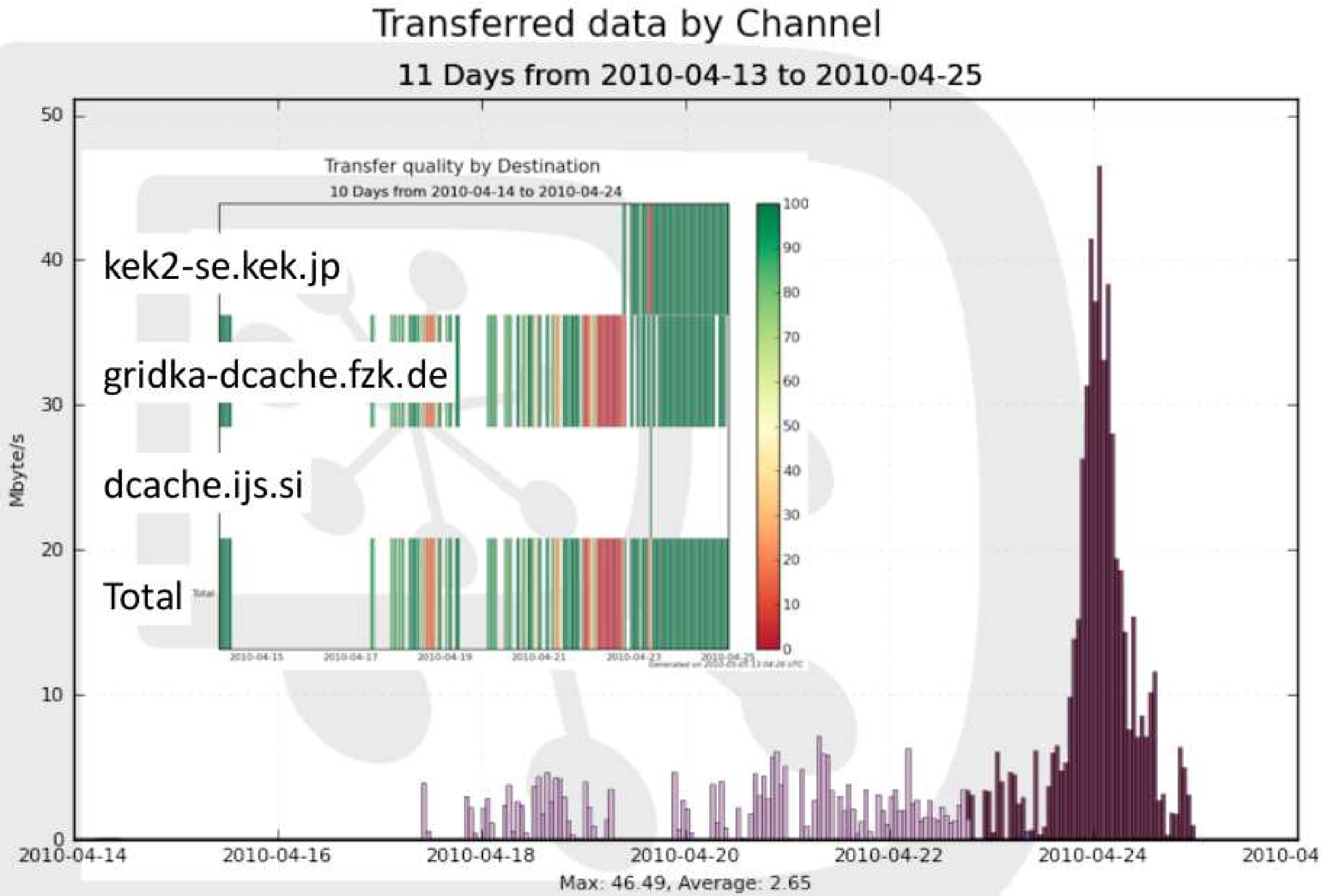
New DIRAC module – Virtual Machine machine manager

Additional 1000 commits to clean up LHCb specifics

DIRAC EC2 -Execution



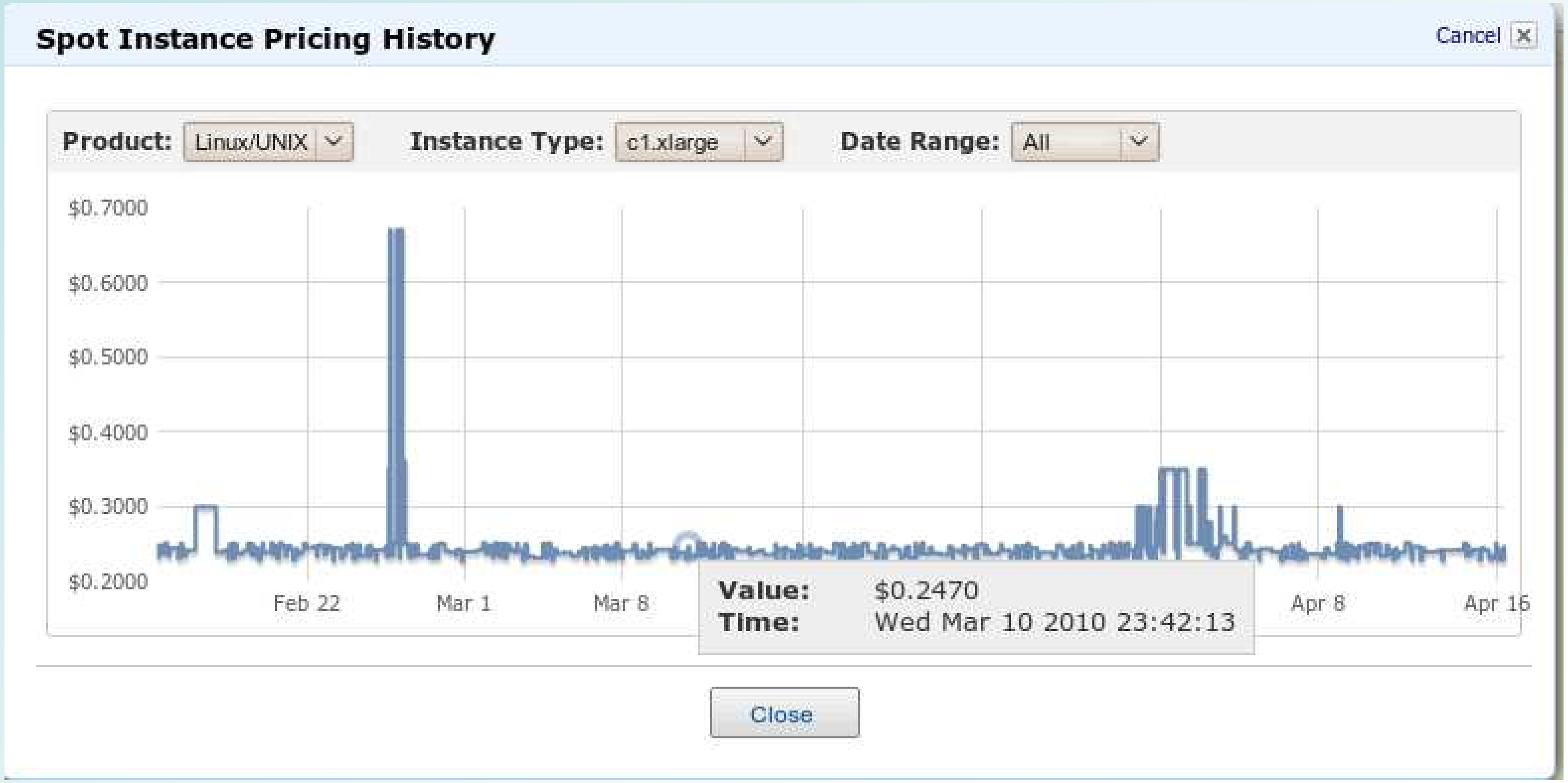
Back to GRID



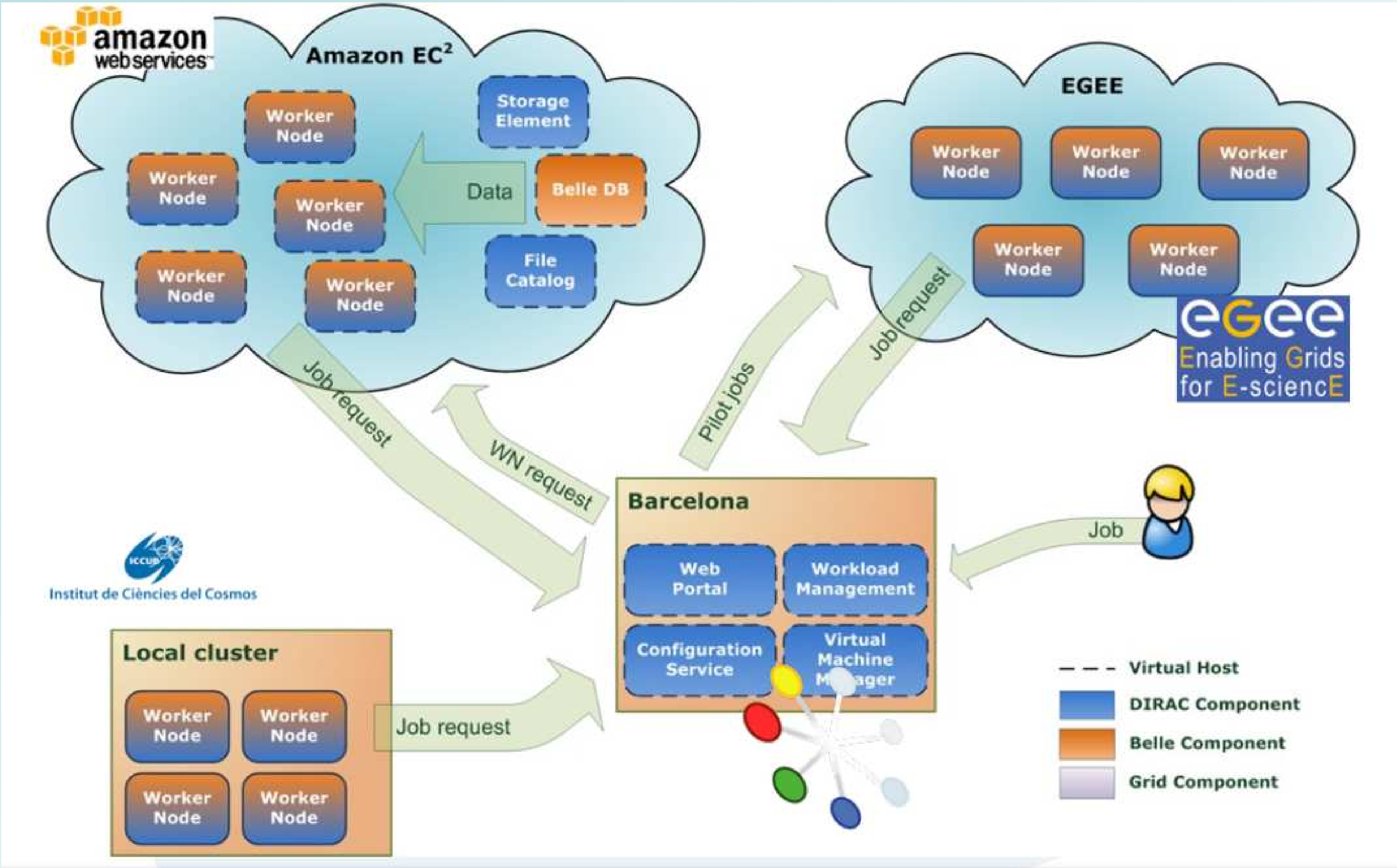
Transferred data by Channel
11 Days from 2010-04-13 to 2010-04-25
Mbyte/s
Transfer quality by Destination
10 Days from 2010-04-14 to 2010-04-24
kek2-se.kek.jp
gridka-dcache.fzk.de
dcache.ijs.si
Total
2010-04-14
2010-04-16
2010-04-18
2010-04-20
2010-04-22
2010-04-24
Max: 46.49, Average: 2.65

EC2 Spot Pricing

- Substantial Reduction in AMI pricing
  - Danger that AMI will be lost during use.
  - ~$0.2 vs. $0.68



Grid, Cloud, Local with DIRAC



Results of DIRAC-EC2 test

- Cloud - Production ready:
  - 5% of Belle production in 10 days
  - 250 M evt (~2.7 TB)
  - In total ~ 4700 CPU days
  - used proven stability and scalability:
  - 2000 CPUs peak achieved in < 4 hours
  - >90% efficiency in CPU usage
- Cost estimation:
  - 0.46 USD/10k evt (reserved price)
  - 0.20 USD/10K evts (Spot pricing)
  - No loss of jobs during spot pricing
  - No admin, cooling and electricity charges
- VWO cost of $4000 server (with electricity) is $0.16 per 10k events
- Input data pre --uploaded to Amazon SE VM.

Remaining issues

- We will require 5,000 – 50,000 cores for a 5 month MC run to match experimental statistics
- Tested 250 instances == 2000 simultaneous cores.
- Can we get good spot prices at this scale?
- Data Retrieval?
- Need to transfer back to GRID at > ~600 MByte/sec
- Multiple SE's to receive data?