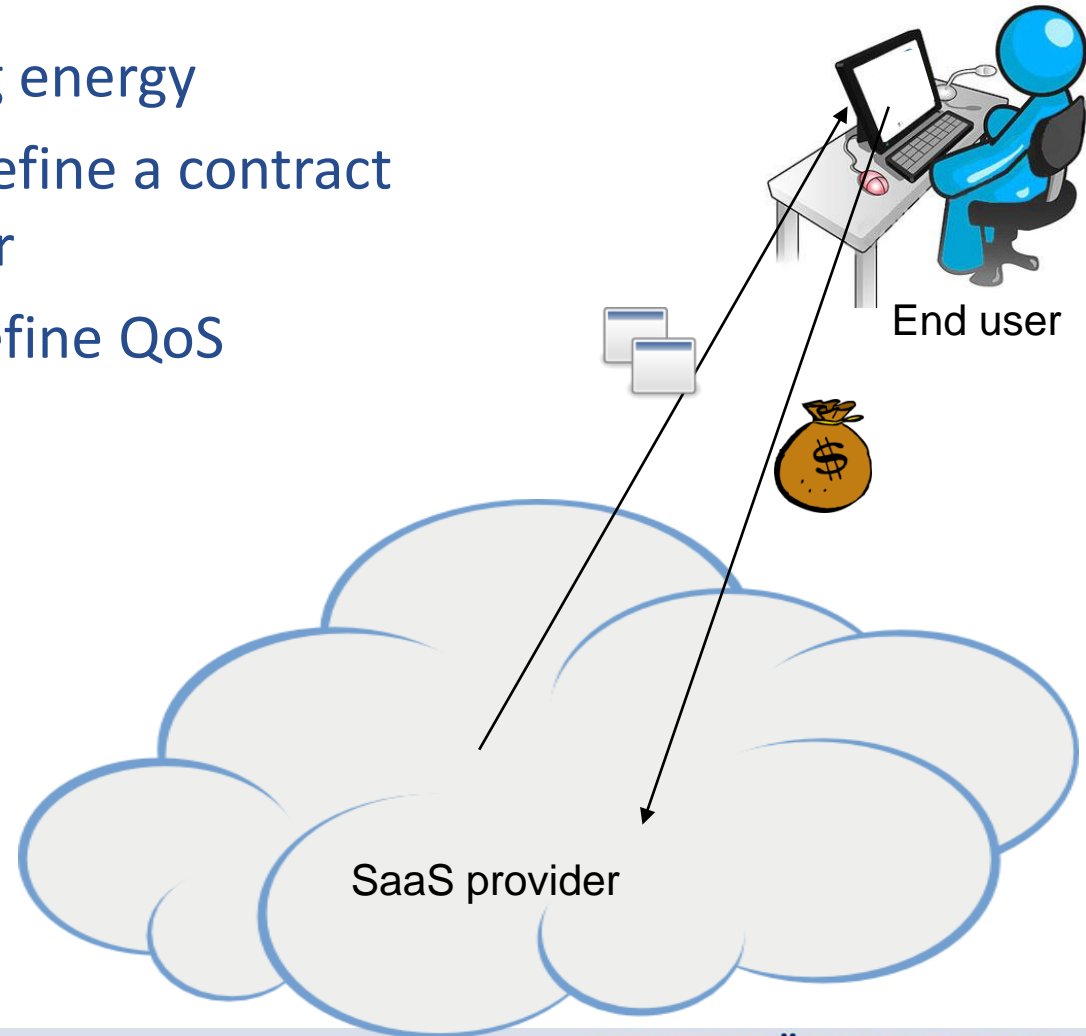


Enforcing SLAs in Scientific Clouds

Oliver Niehörster, André Brinkmann,
Gregor Fels, Jens Krüger, and Jens Simon

Software as a Service SaaS

- On demand, pay-as-you-go
- Consume software like receiving energy
- Service Level Agreement SLAs define a contract between provider and consumer
- Service Level Objectives SLOs define QoS requirements inside SLA
- Business examples:
 - Word Processor, CRM, CAD, Spreadsheet, ...



Challenges

scientific SW vs. business SW

- One time jobs
- SLO: performance or deadline
- Performance depends on job inputs
- measuring the QoS is difficult

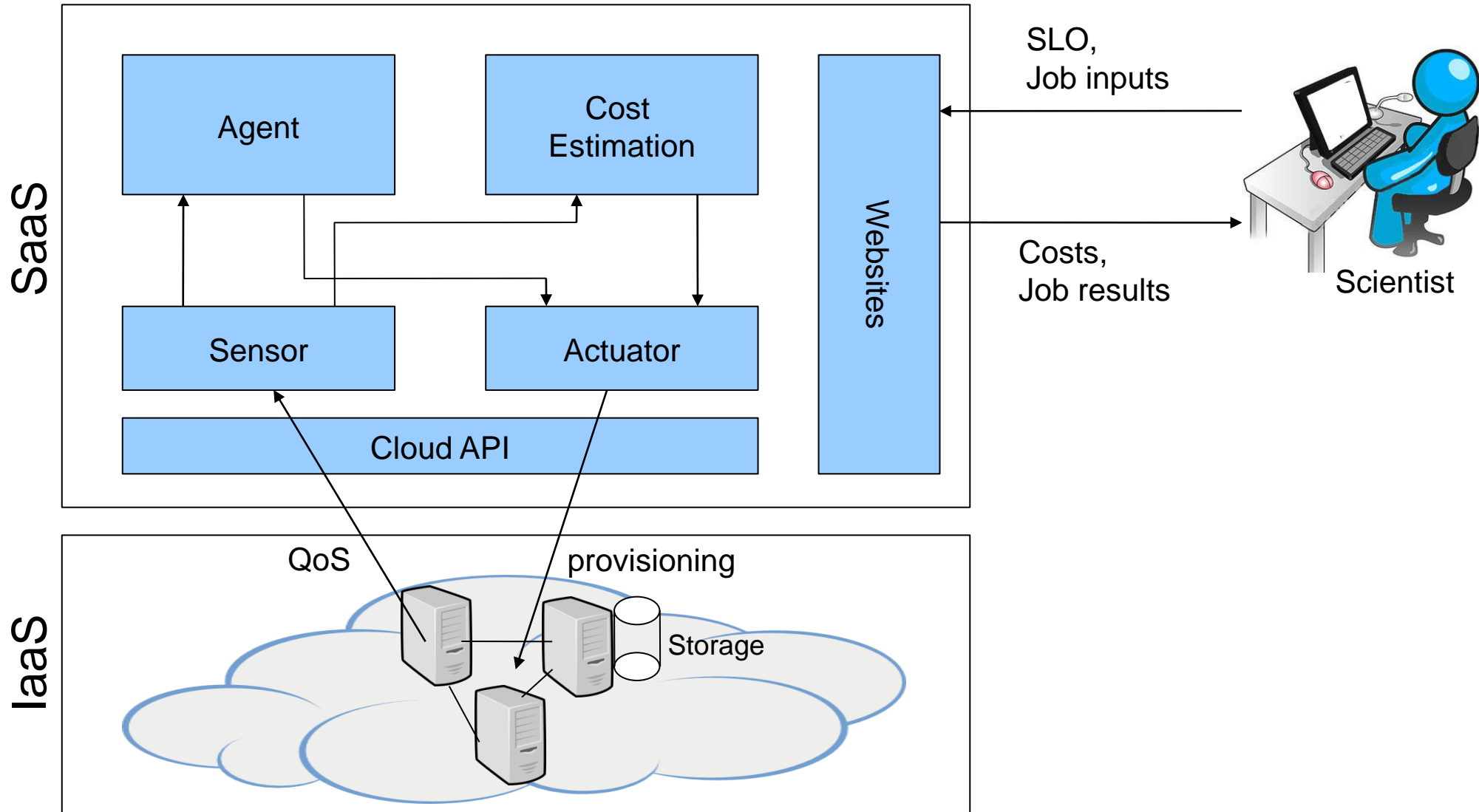
A cloud is ...

- private or public,
- a blackbox,
- used by many users.

- 24/7 services
- SLO: availability, requests/sec
- QoS data mostly available, directly in logfiles or via benchmarking

- Enforce SLAs in a stochastic environment, where VMs ...
- influence each other,
 - compete against resources

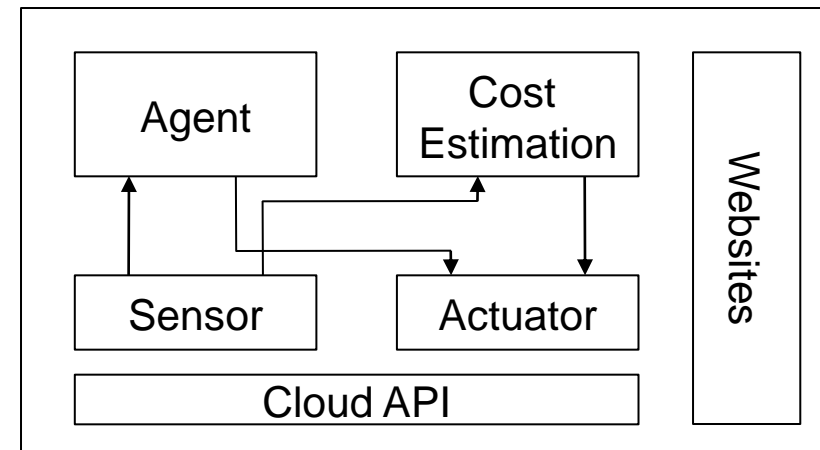
Architecture



Sensors and Actuators

- **Sensors:**

- read environmental information
- monitoring data
- read log files
- run benchmarks
- application specific

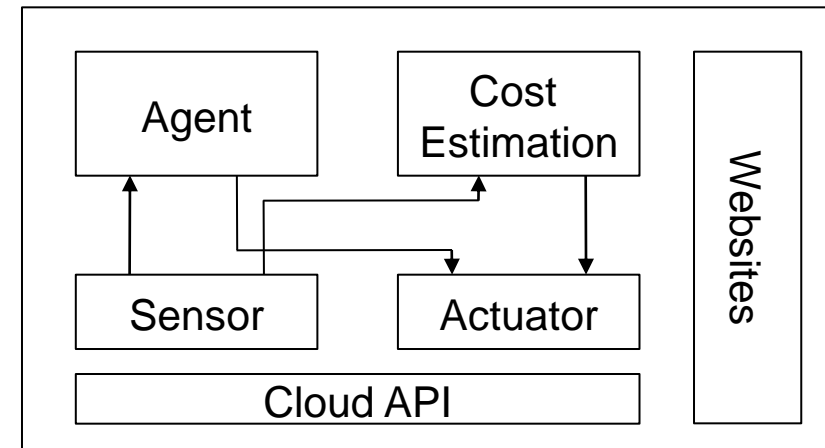


- **Actuators:**

- influence the environment
- create, configure, setup, change, terminate instances
- are application specific (especially setup and configuration)

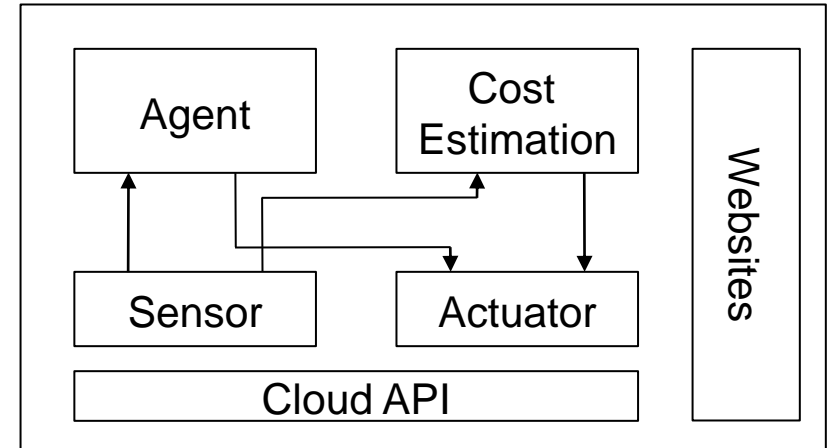
Cost Estimation (CE)

- Inputs: job data, SLO
- Outputs:
 - cost estimate
 - minimal resources (if SLO is feasible)
- Possible approaches:
 - heuristic based on previous job runs
 $h(\text{job inputs}) = \text{needed resources}$
 - pilot runs: do a search with short representative jobs



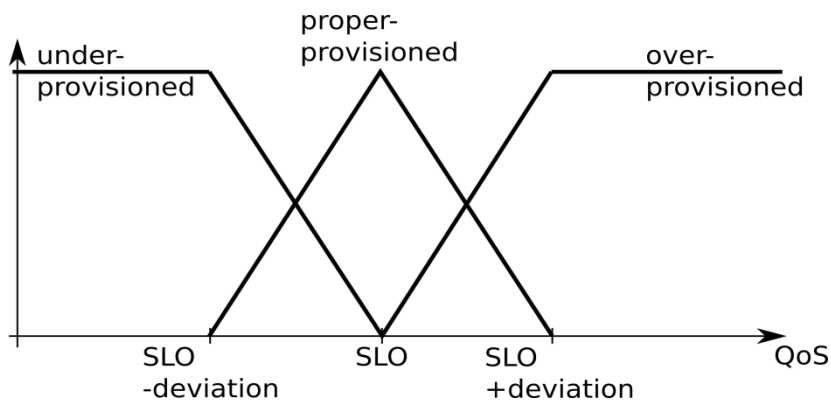
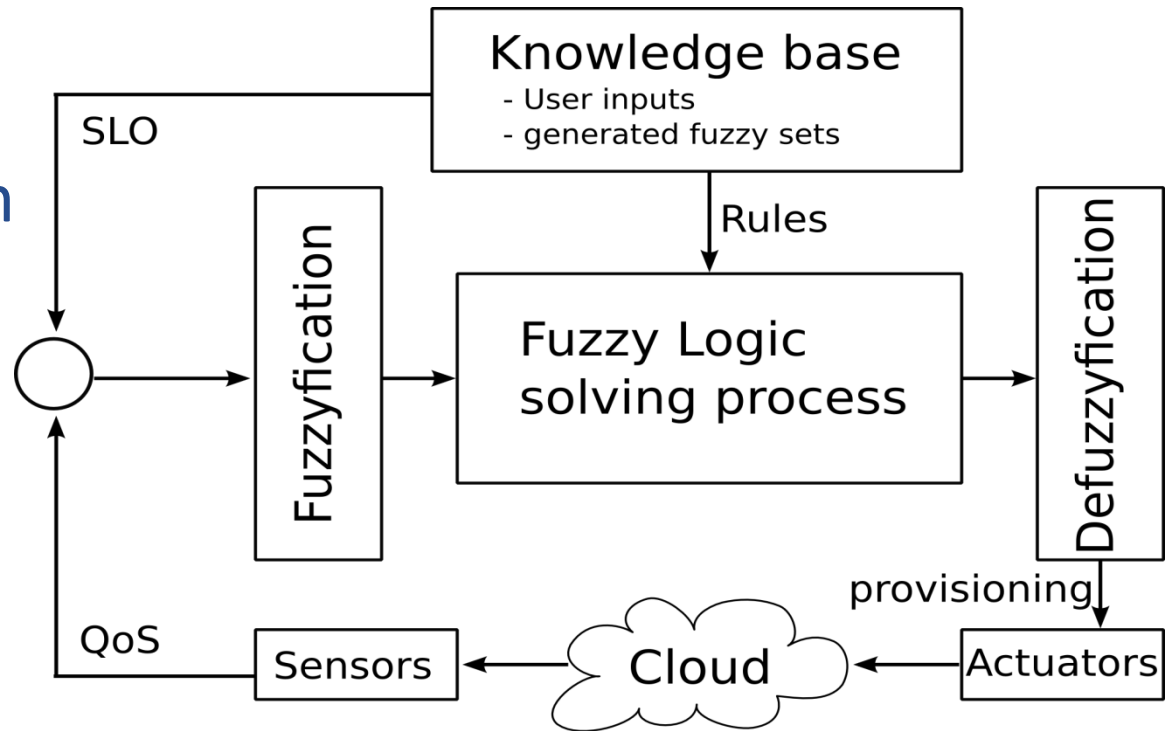
Agent

- One agent per job that has to enforce SLOs
- Skills:
 - Determine the “optimal” provisioning that fulfills the SLO
 - Can change node type
 - Can change number of nodes
- Behavior:
 - Read sensor data
 - Assess QoS
 - Correct provisioning with the actuators



Agent

- Reflex agent
- Fuzzy control system
- “soft” SLO as performance contract



IF under-provisioned

THEN faster

IF proper-provisioned

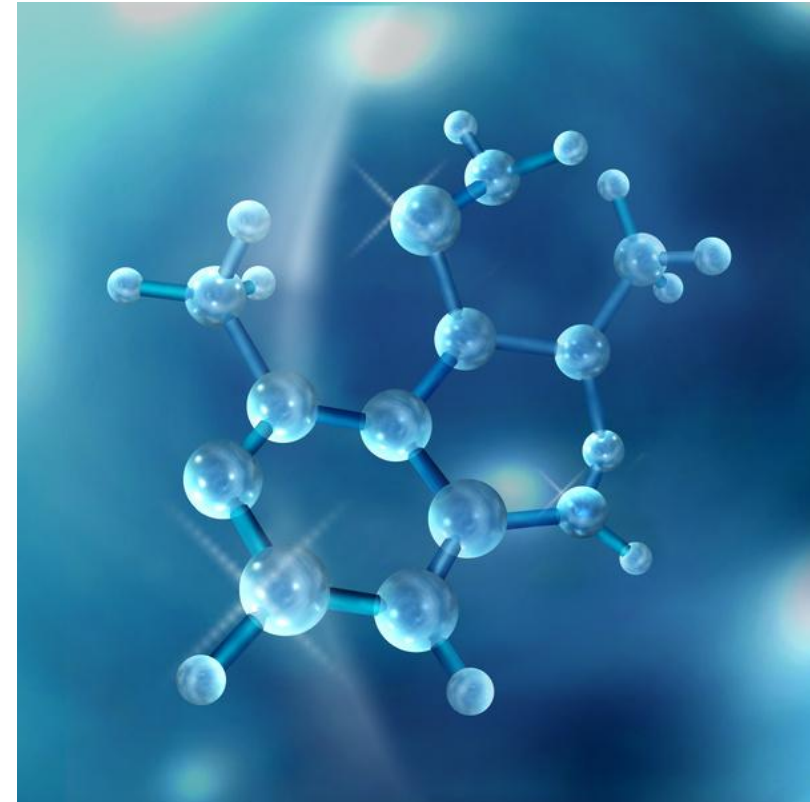
THEN no change

IF over-provisioned

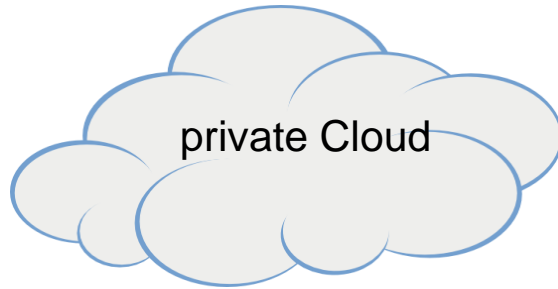
THEN slower

Gromacs as a Service (GraaS)

- Molecular dynamics package designed for biomolecular systems
- SLOs: deadlines
- Sensor:
 - Reads Gromac's time estimates from stdout
- Actuators:
 - Adding and removing of nodes
 - Reconfiguration of the applications (assure that #MPI processes = #vCores)
 - Configure and restart via checkpoints
 - Change type first, then change #VMs
- Cost Estimation:
 - Binary search
 - Pilot runs with the original job data but with fewer iterations

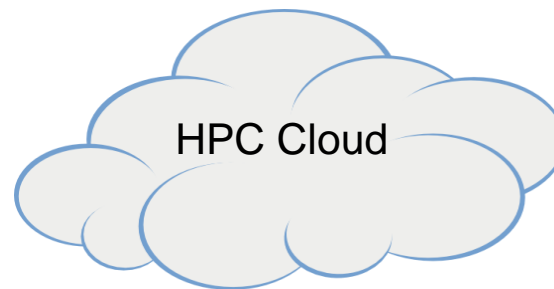


Evaluation



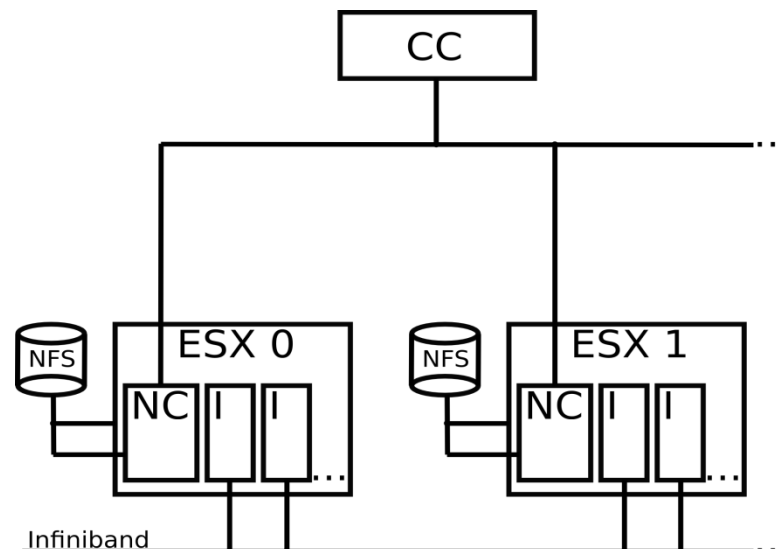
private Cloud

- Eucalyptus
- EC2 interface
- XEN nodes
- #vCPUs ≤ #pCores
- Change instance type
 - Ballooning
 - CPU hotplug



HPC Cloud

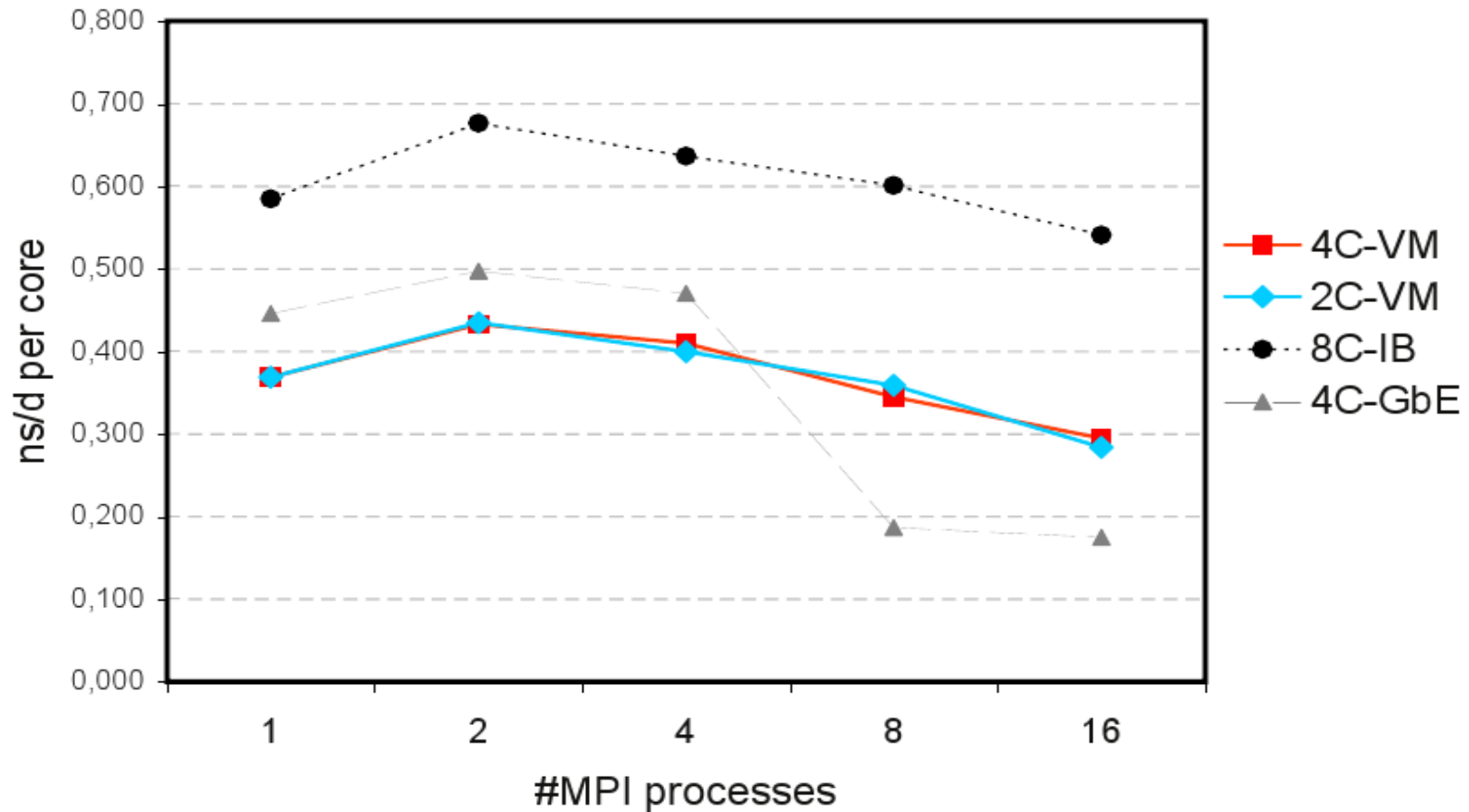
- Like the private cloud
- ESX nodes with Infiniband HCAs



Amazon's EC2

- Public cloud
- m1.large (4 EC2 CUs) in us-east-1a
- Ubuntu instances
- Just horizontal scaling

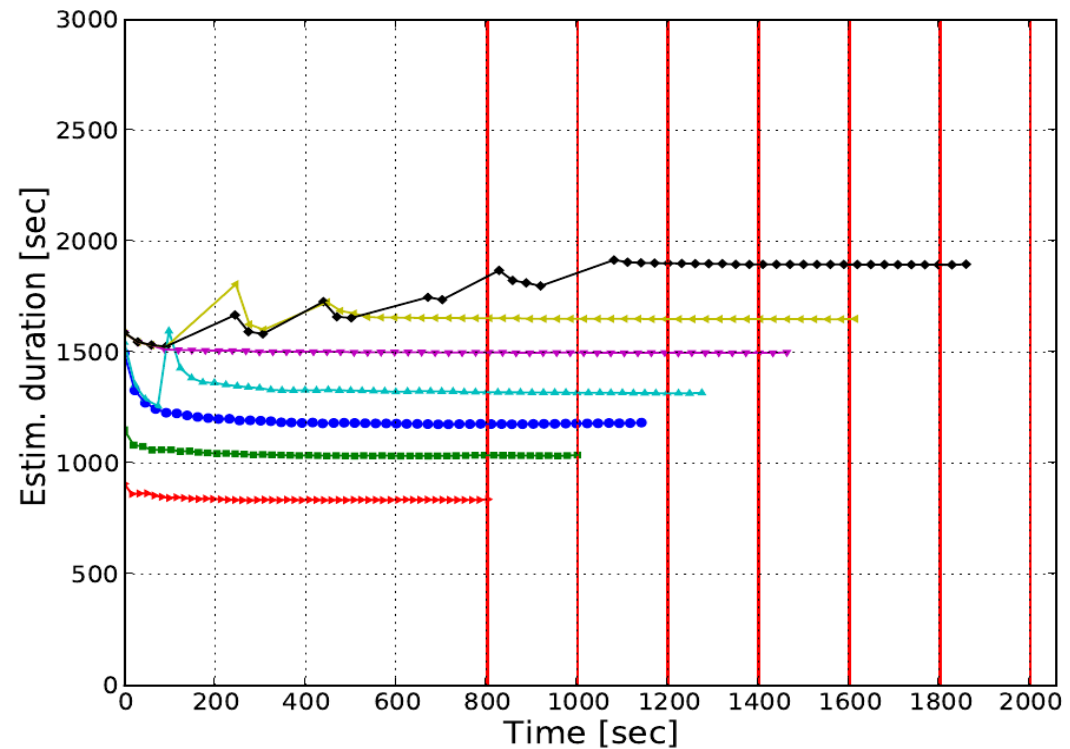
- Performance evaluation with Gromacs (DPPC in water)



Evaluation

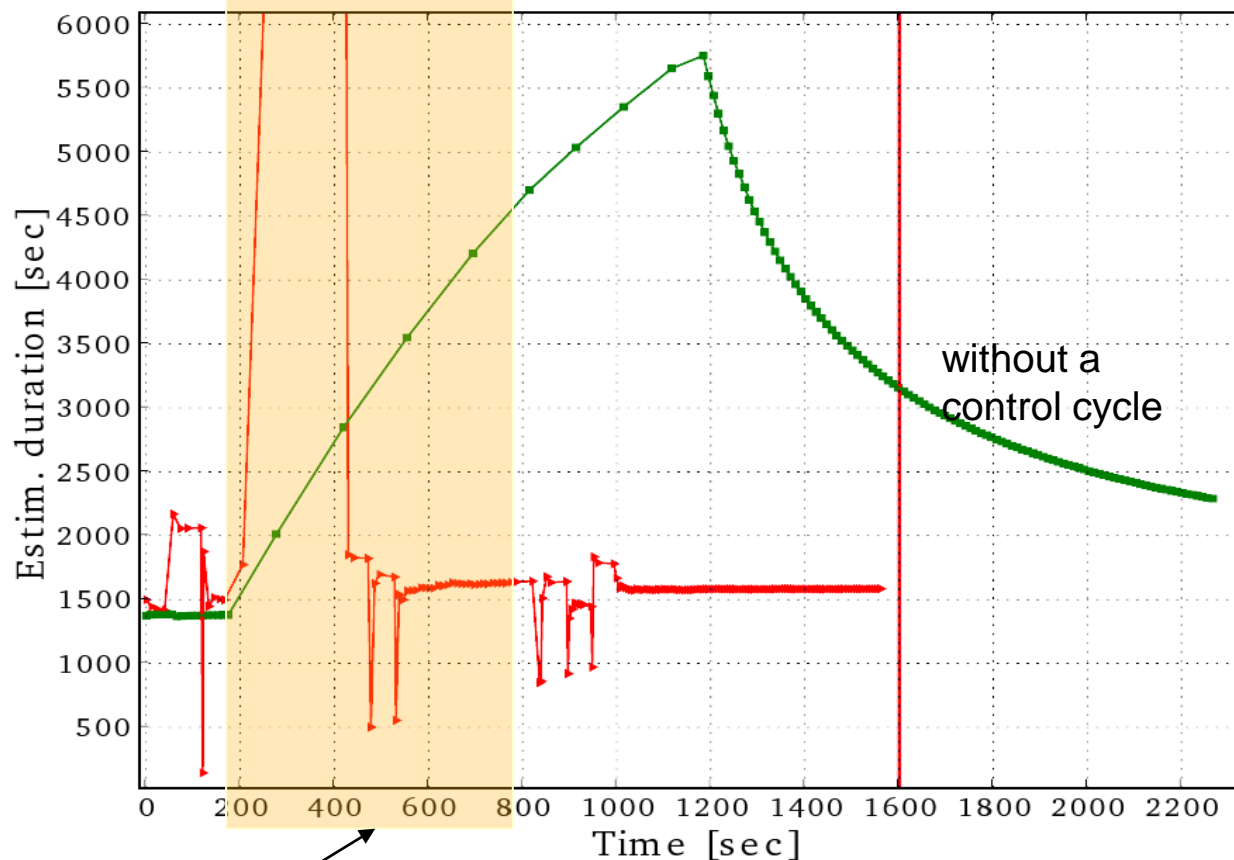
- Private Cloud
- Empty: no noise
- Job: *DPPC in water*, 12.500 iterations

Deadline [sec]	CE time [sec]	CE #nodes	real end [sec]	real costs [credits]
2000	1508	1	1859	84
1800	1508	1	1614	77
1600	1500	1	1463	73
1400	1240	3	1275	130
1200	1210	3	1142	171
1000	1044	4	1001	200
800	812	6	802	240



Evaluation

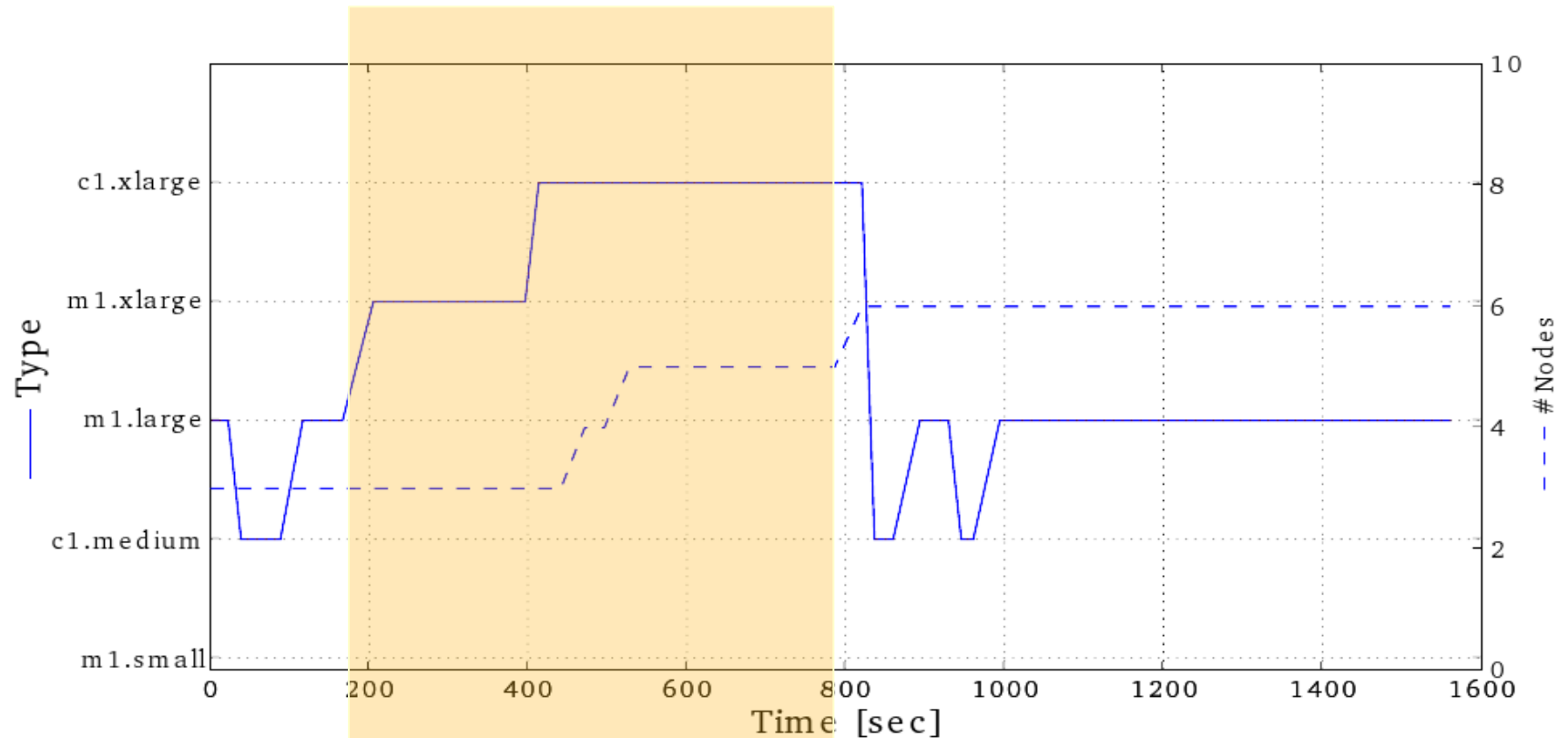
- Two jobs in a noisy cloud
- Synthetic disk, I/O, and memory load on the hypervisors



load

Evaluation

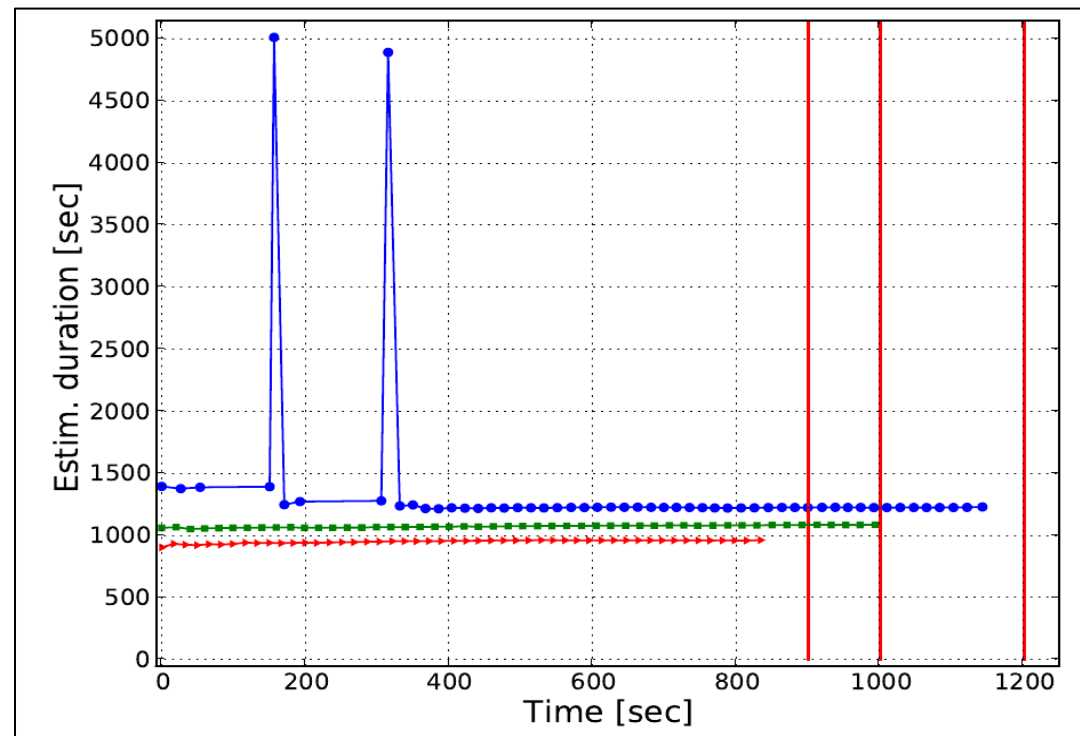
- Two jobs in a noisy cloud
- Synthetic disk, I/O, and memory load on the hypervisors



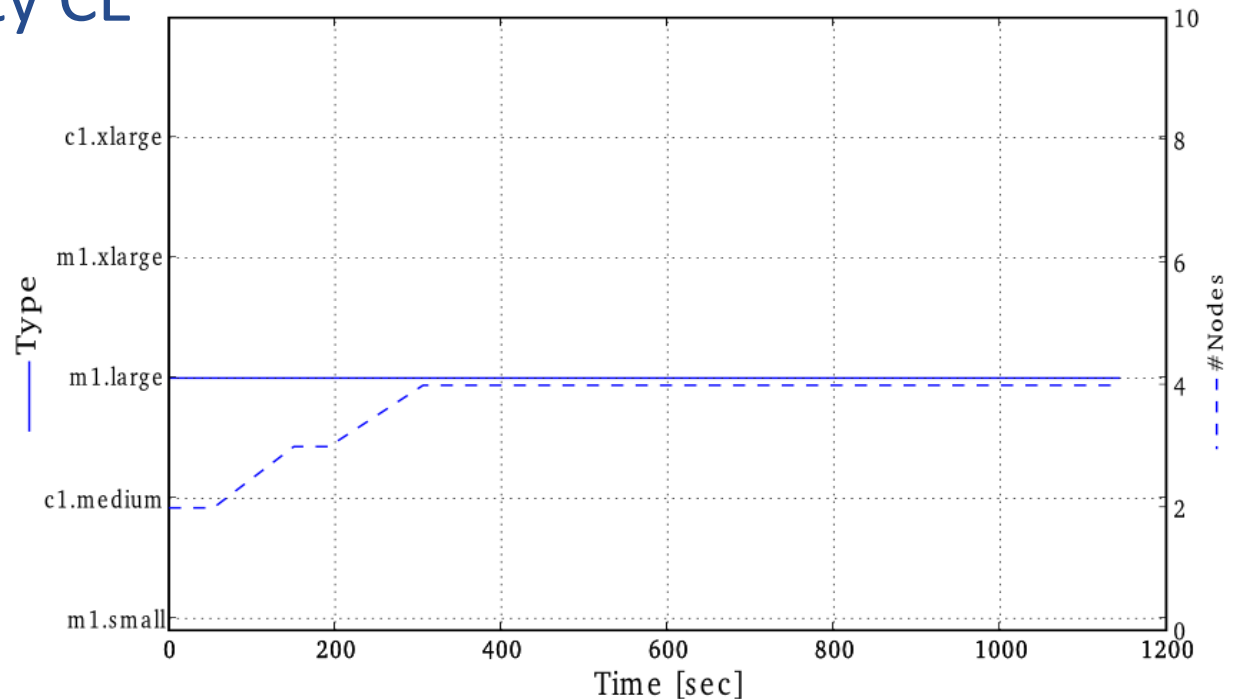
load

Evaluation

- EC2 observations:
 - Performance of one instance relative stable
 - But differences between identical instances
- Starting with a faulty CE



- EC2 observations:
 - Performance of one instance relative stable
 - But differences between identical instances
- Starting with a faulty CE



Summary and Future Work

- Scientific SaaS with an easy-to-use website
 - Works with applications that are malleable
 - Provides an interface for QoS measuring
 - Allows a cost estimation
 - Agents:
 - Application specific
 - Control the jobs
 - Enforce SLA
 - Cost Estimation
 - Evaluation:
 - GraaS on three different clouds: private, private HPC, EC2
 - Benchmarks of the HPC cloud
- Future Work:
 - More applications
 - Smarter agents, who cooperate instead of compete
 - Handle “finite” clouds

Thank you!