The Green Computing Observatory: status of acquisition and analysis

Cécile Germain-Renaud[1], Julien Nauroy[1], Michel Jouvin[2], Guillaume Philippon[2]

1: Laboratoire de Recherche en Informatique, U. Paris Sud, CNRS, INRIA

2: Laboratoire de l'Accélérateur Linéaire, CNRS-IN2P3

- Previous *GreenDays* talks
  - *GreenDays@Paris* The Green Computing Observatory: plans and scientific challenges
  - *GreenDays@Lyon* The Green Computing Observatory: from instrumentation to ontology

- This talk
  - General objectives
  - Acquisition: what's new
  - Analysis: some simple models

A Grid Observatory project

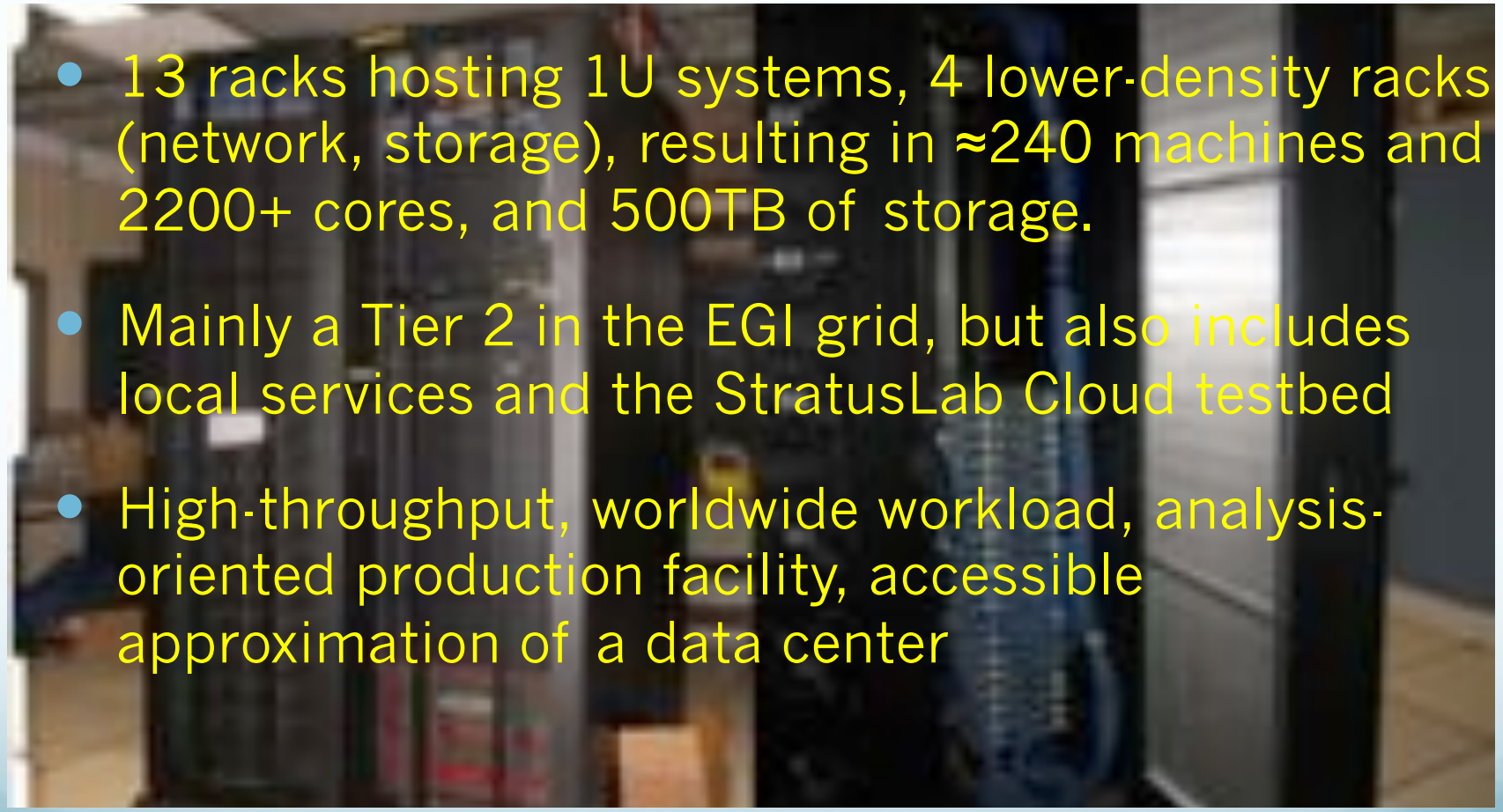
A Digital Curation approach

- Establish long-term repositories of digital assets for current and future reference: continuously monitoring a large computing facility
- Tackling the good data creation and management issues, and prominently interoperability: formal mainstream ontology, standards-aware
- Providing digital asset search and retrieval facilities to scientific communities through a gateway
  - Files in XML format
  - Available from the Grid Observatory portal

With the support of/supporting

- France Grilles – French NGI member of EGI
- EGI-Inspire (FP7 project supporting EGI)
- INRIA – Saclay (ADT programme)
- CNRS (PEPS programme)
- University Paris Sud (MRM programme)

The GRIF-LAL computing room



- 13 racks hosting 1U systems, 4 lower-density racks (network, storage), resulting in ≈240 machines and 2200+ cores, and 500TB of storage.
- Mainly a Tier 2 in the EGI grid, but also includes local services and the StratusLab Cloud testbed
- High-throughput, worldwide workload, analysis-oriented production facility, accessible approximation of a data center

Sensors

Grid Observatory

Smart meter

PDU

Twin² server

IPMI

Machine

Ganglia

Processor

4 cores

2-4 processors

4 machines

220 machines

2GBytes/day at 1 minute sampling period

Sensors

Grid Observatory

Smart meter

PDU

Twin² server

IPMI

Machine

Ganglia

Processor

4 cores

2-4 processors

4 machines

220 machines

2GBytes/day at 1 minute sampling period

3GBytes/day total

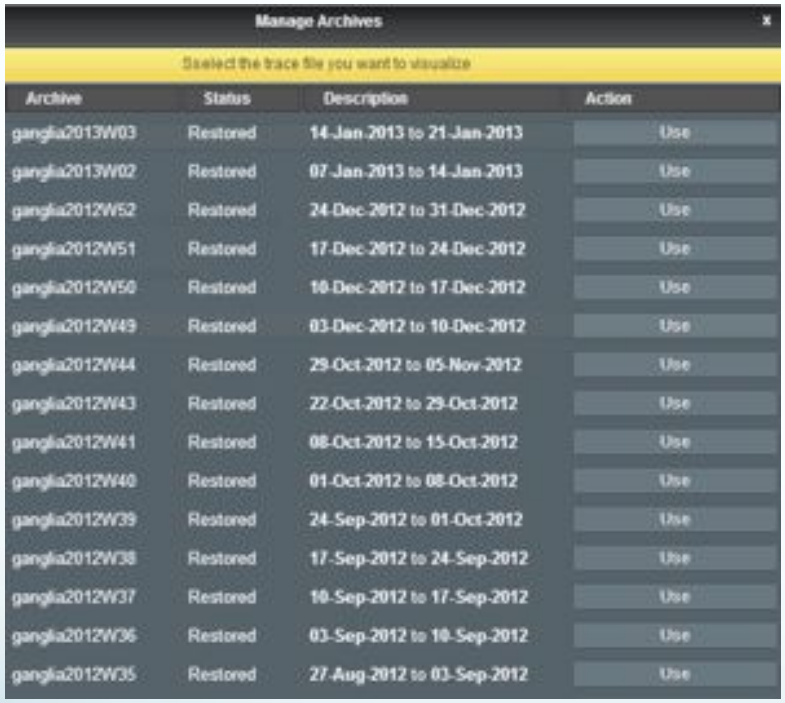
What's new

- Improving the user's Quality of Experience: visualization through the GCO Monitor
- Room power acquisition
- Monitoring a Cloud
- Internals: we went « push » thanks to the ActiveMQ protocol

The GCO Monitor

1. Choose acquisition Week

2. Browse and visualize



Room power consumption

Grid Observatory

Acquiring data from the 3-phase power supply

- Data is acquired every ~12s
- Measurements:
  - Voltages & Intensities (3 + 3)
  - Harmonic distortions
  - Current load

- Curation at its best
  - Harmonic distortions due to computer PSUs
  - Unbalanced load over phases
  - Load includes computers, CRACs, network equipments, light...

Monitoring the Stratuslab cloud

- IaaS Cloud based on OpenNebula
- FP7 project, complete. Two reference infrastructures at GRNet (256 cores) and LAL
- Stratuslab@LAL, as of January 2013
  - 240 cores total, 1 head node, 10 hosts for VMs, 24 CPU, 32GB RAM per host + 20 TB storage space
  - ~30 new VM per day
  - EGI and Stratuslab clusters are on separate networks
- Interest: testing the concepts on a standard-based development, and be ready for monitoring production

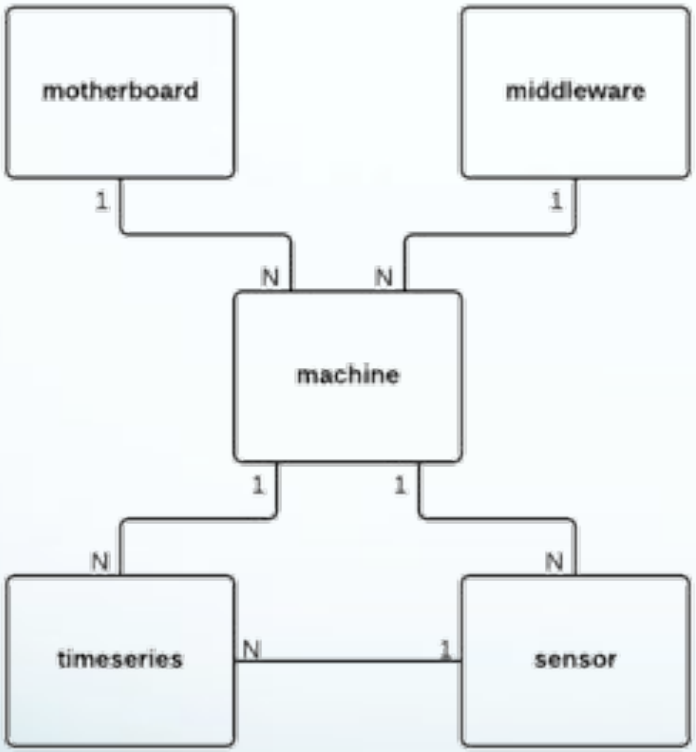
Brief recap: why use ontologies in GCO?

- Our purpose
  - To clarify the *semantics* of data
    - To get a *computational* model
  - To define an *ontological semantics* for the XML schema
- Our approach
  - To define a *semantically transparent* ontology
    - To reuse the *foundational* DOLCE[1] ontology
  - To use the OntoSpec[2] methodology (**modularity + high expressiveness**) which integrates the OntoClean[1] methodology

[1] Laboratory for Applied Ontology: http://www.loa.istc.cnr.it/

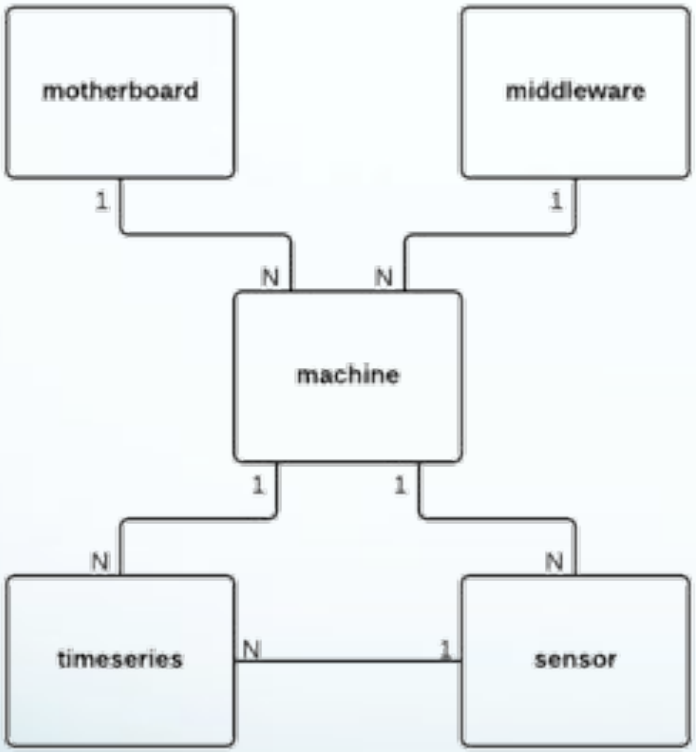
[2] http://home.mis.u-picardie.fr/~site-ic/site/?lang=en

Translated into XML format



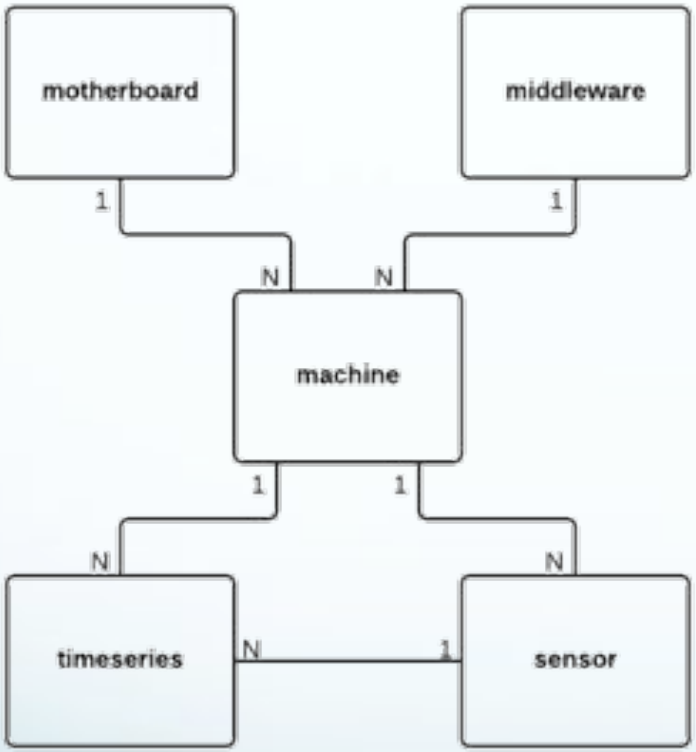
- Where should we put hypervisors: “Machines” or “Middleware”
  - The machine concept is the entity supporting computation
    - Hardware + OS for non-virtualized exploitation
    - Hardware + Hypervisor for virtualized exploitation
- Thus the Middleware concept

Translated into XML format



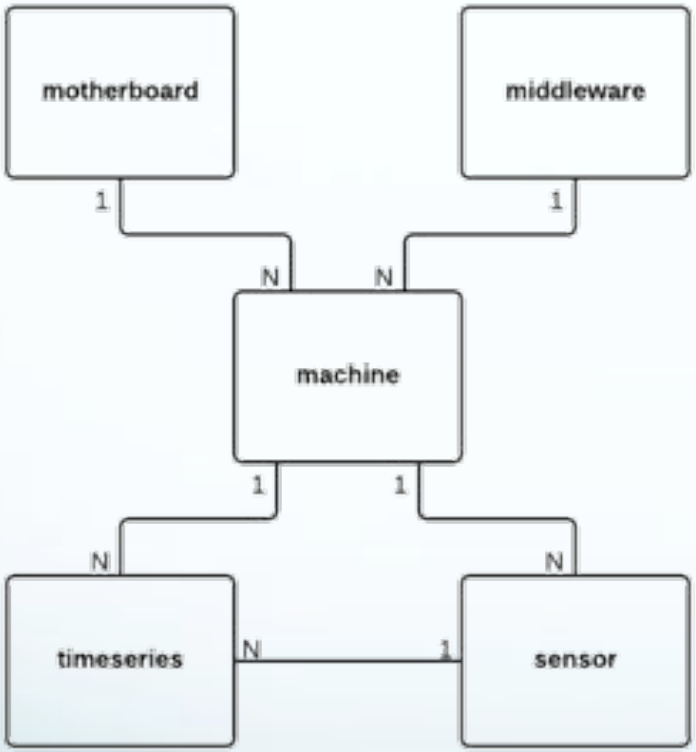
- Where should we put Virtual Machines: “Machines”, “Middleware”or “Timeseries”?

Translated into XML format



- Where should we put Virtual Machines: “Machines”, “Middleware”or “Timeseries”?
- The specific conceptualization revolves over a temporal characterization
  - Identity properties: never change, e.g. motherboard information, middleware version
    - Used to fully identify the entity
  - Slowly mutable properties: can sometimes change, e.g. IP address, Firmware version
    - Keep a history of the modifications
  - Time series: values constantly change e.g. CPU activity, power consumption
    - Keep track of every value and acquisition date

Translated into XML format



- Where should we put Virtual Machines: “Machines”, “Middleware”or “Timeseries”?
- The specific conceptualization revolves over a temporal characterization
  - Identity properties: never change, e.g. motherboard information, middleware version
    - Used to fully identify the entity
  - Slowly mutable properties: can sometimes change, e.g. IP address, Firmware version
    - Keep a history of the modifications
  - Time series: values constantly change e.g. CPU activity, power consumption
    - Keep track of every value and acquisition date

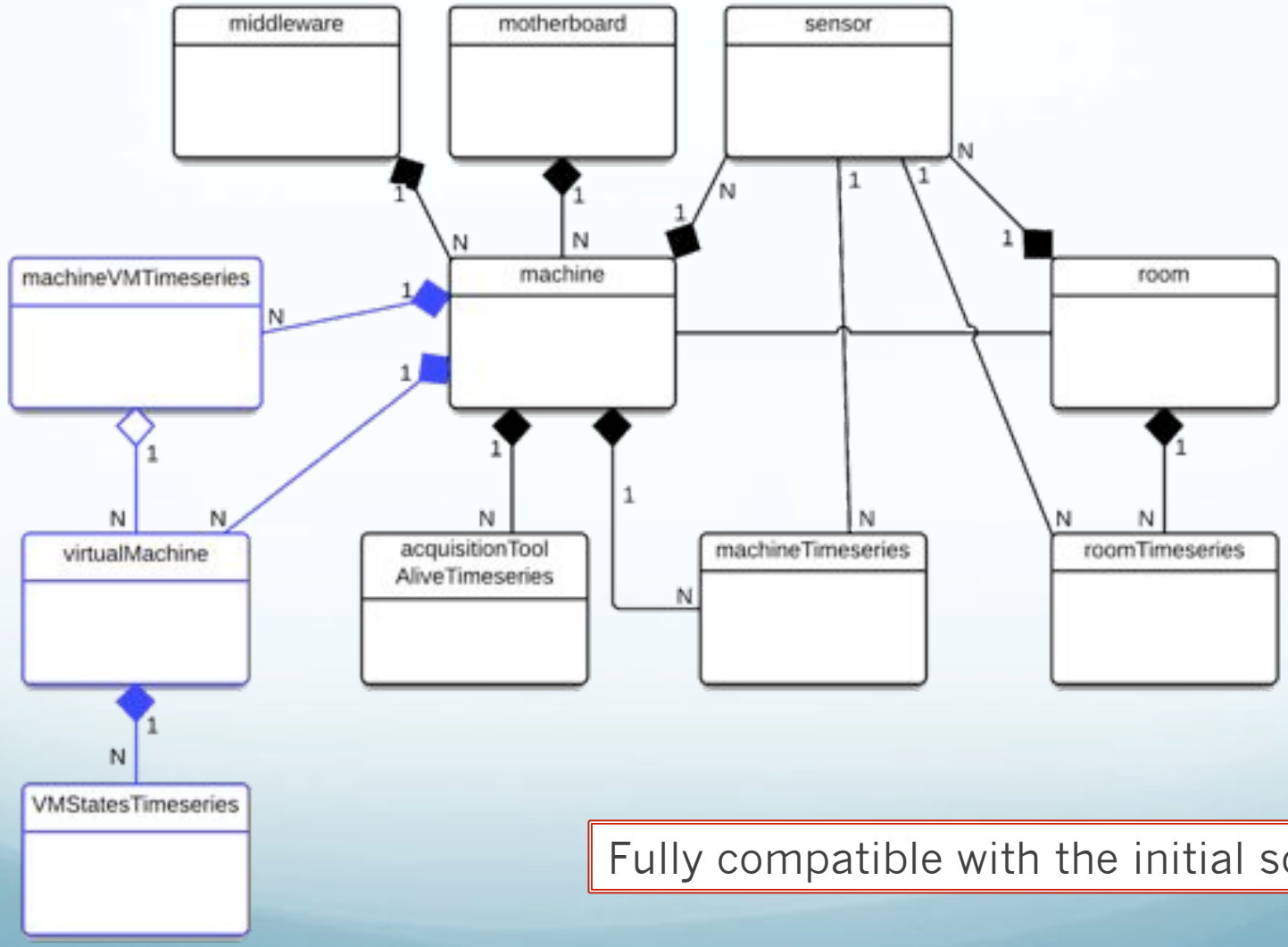
Thus the answer is Timeseries

Virtual Machine data

- Hypervisor the VM is dispatched onto
- Life cycle (starting, running, shutdown, unknown, etc)
- Required resources (CPU, RAM)
- Virtual image from marketplace (os name & version, arch)
- Other: user, IP address, MAC address...

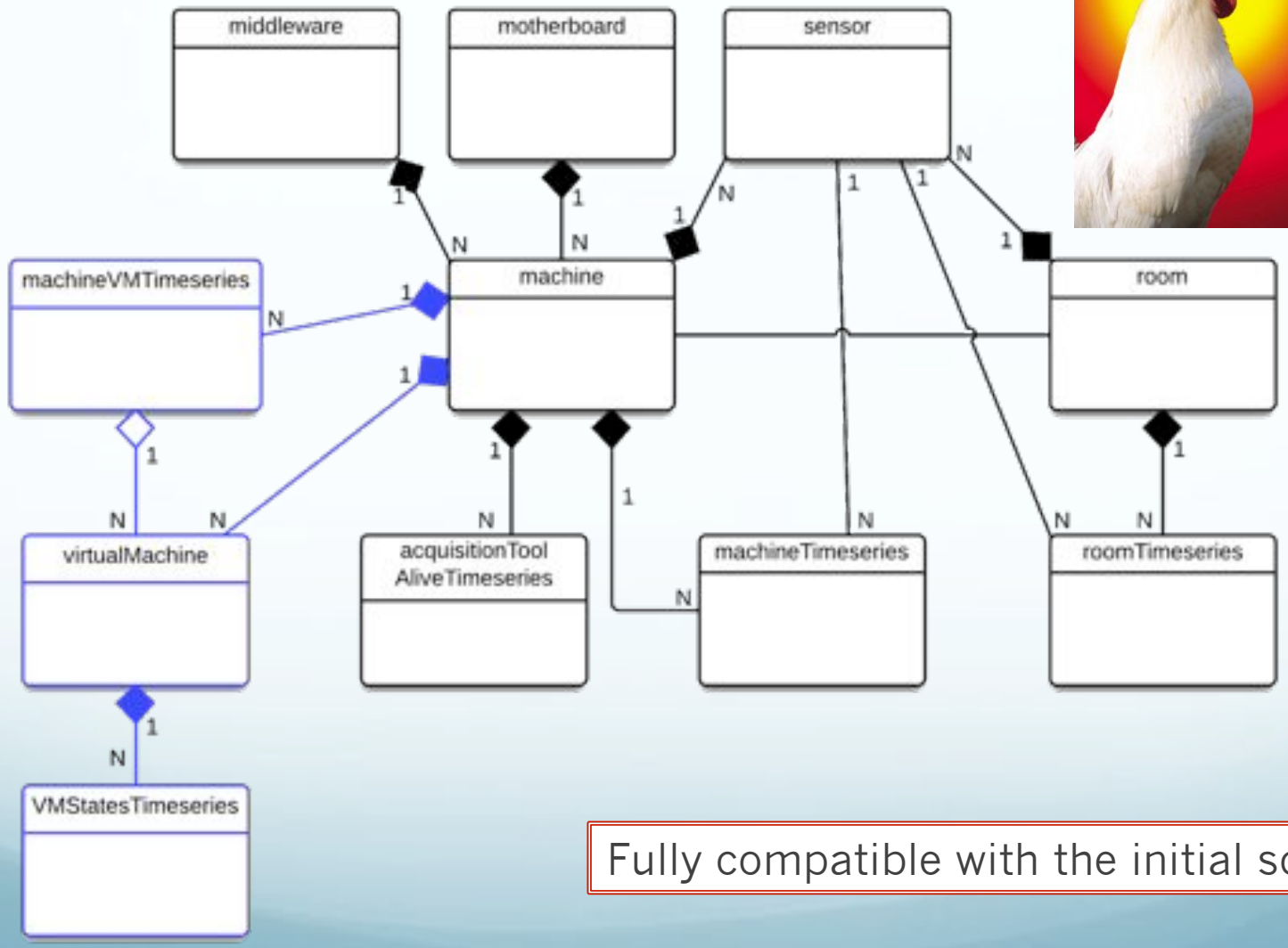
```
localID => 2187
userID => 8, userName => JulienNauroy
hostName => onehost-19.lal.in2p3.fr, hostMachineID => 876
resources =>
    memory => 1024
    CPU => 2, VCPU => 2
    image => HZTKYZgX7XzSokCHMB60lS0wsiv, middlewareID => 14289
states =>
    1357893510 => ACTIVE
    1357893510 => PROLOG
    1357893528 => BOOT
    1357893529 => RUNNING
    1357909600 => SHUTDOWN
    1357909606 => EPILOG
    1357909613 => DONE
```

XML schema status



Fully compatible with the initial schema

XML schema status

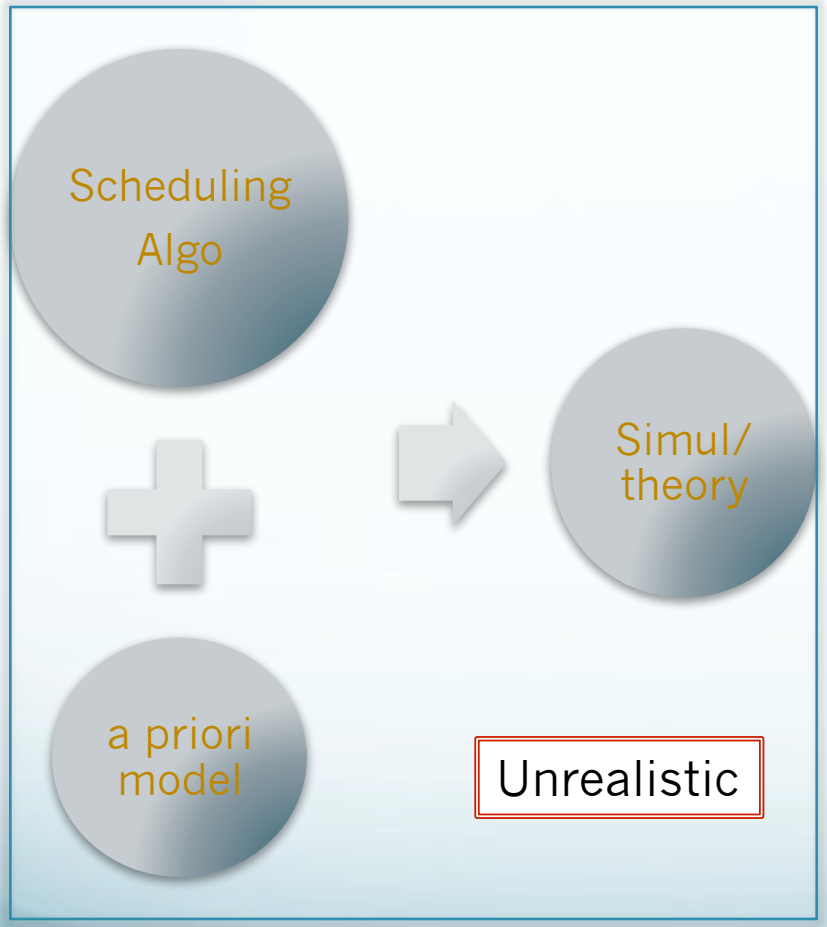


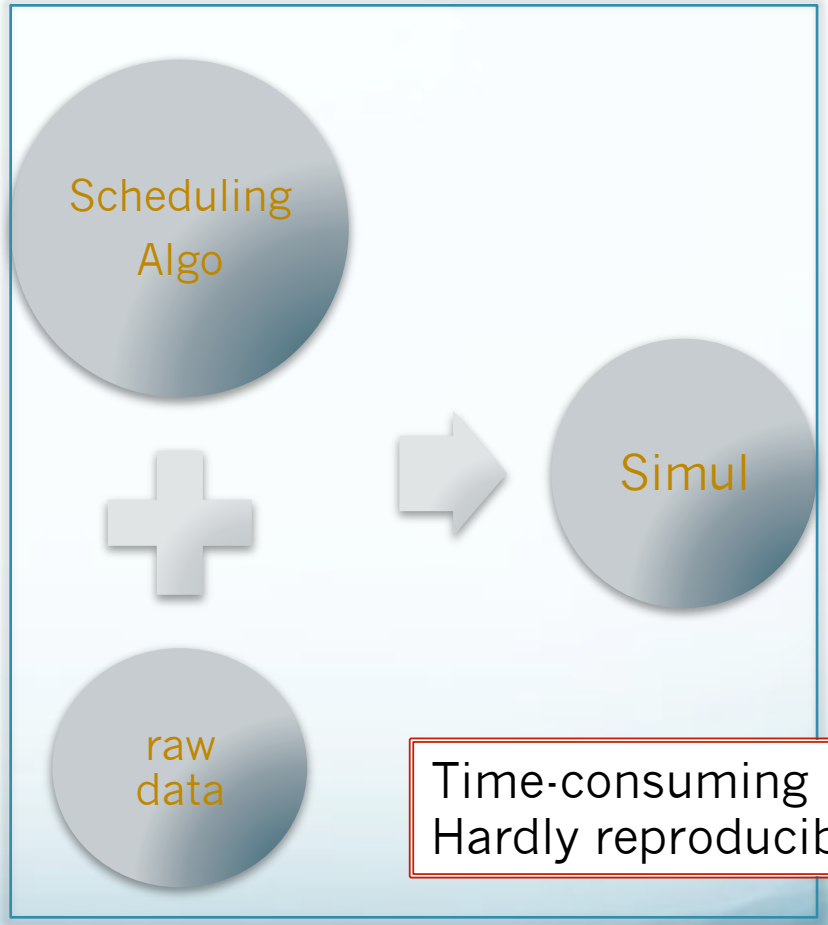
Fully compatible with the initial schema

Metrics, Measures and Models from GreenDays@Paris

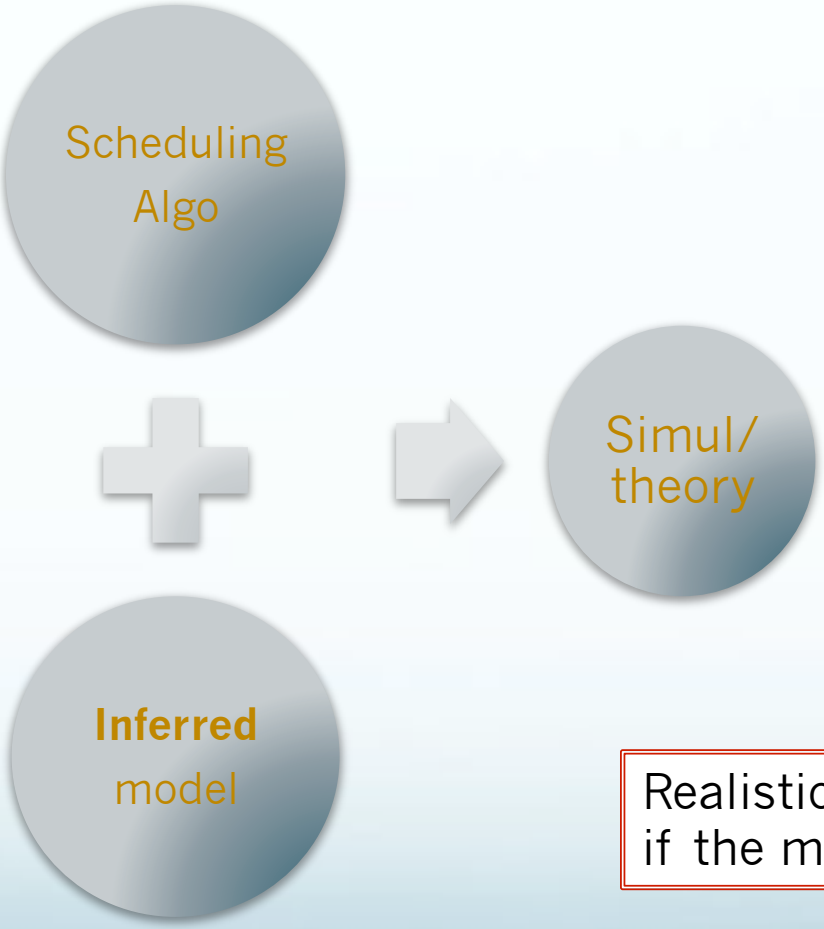
- First step: behavioral descriptive models i.e. parsimonious representations from the large dimension space available from the detailed monitoring
  - Stationarity should not be assumed -> detection of ruptures
  - On-line, dynamic clustering with GStrAP
- Next: identify optima in the resulting complex landscape
- Requires the developement of a framework for automated analysis, in particular data correlations/clustering
- 200+ systems!

Models and validation





Models and validation



Realistic and reproducible
if the model is credible

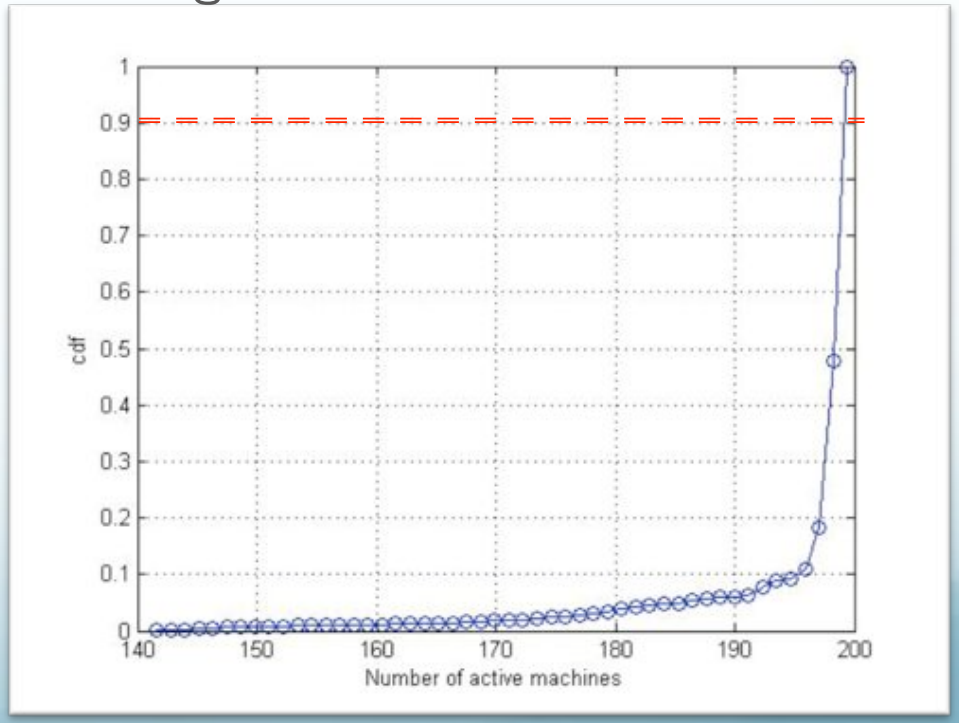
Experiment

- 1-29 February 2012
  - 202 machines
  - Nearly nominal

    BUT shutdowns, monitoring failures etc. have to be taken into account

Active at t = readable ganglia data

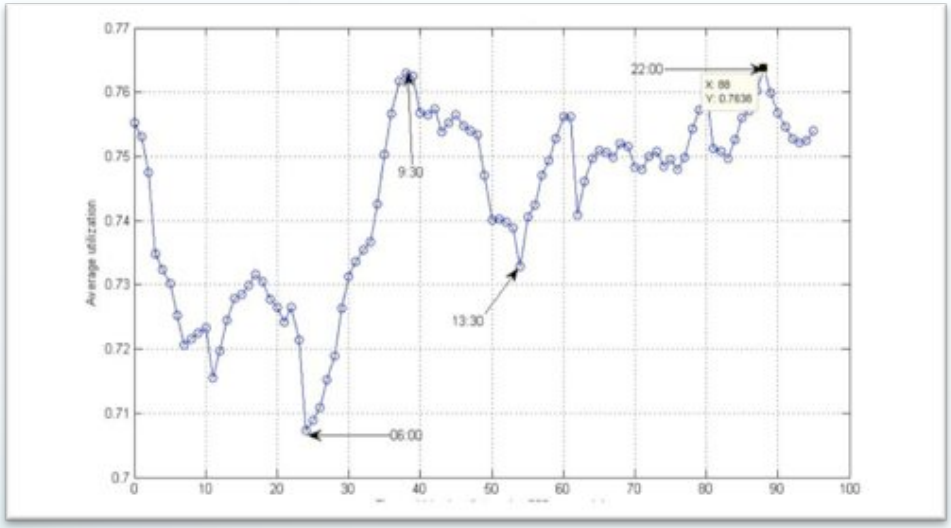
IPMI data may not be available

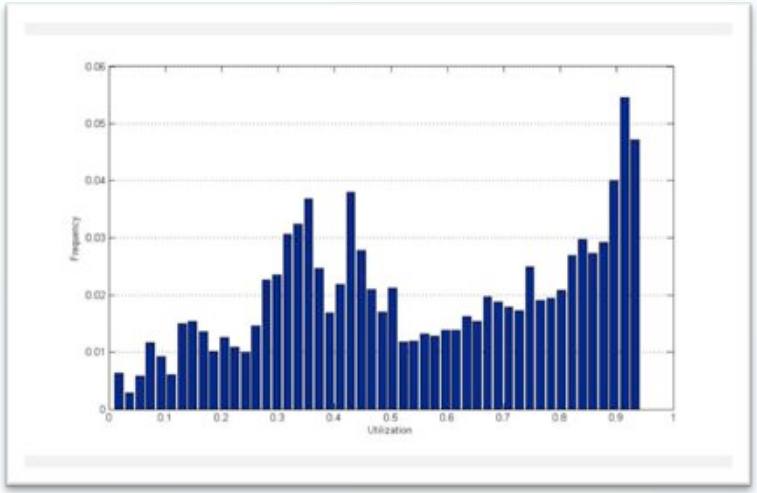


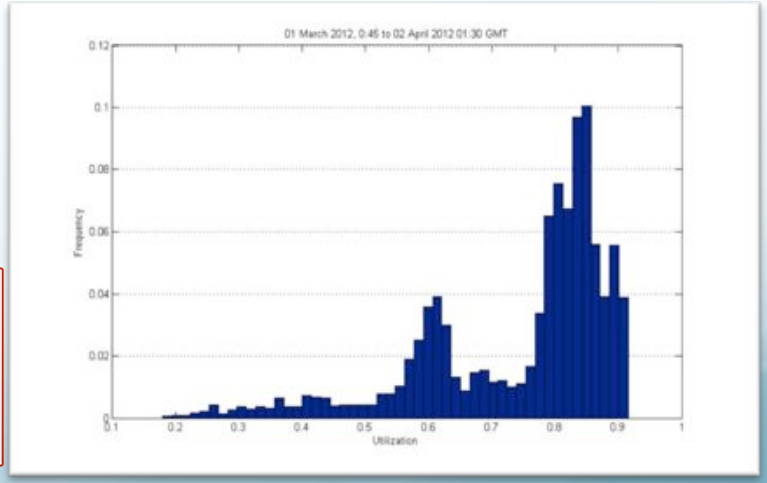
Descriptive statistics

Grid Observatory

Average at constant day time







- Seasonality
- A more serious issue: are our systems stationary?

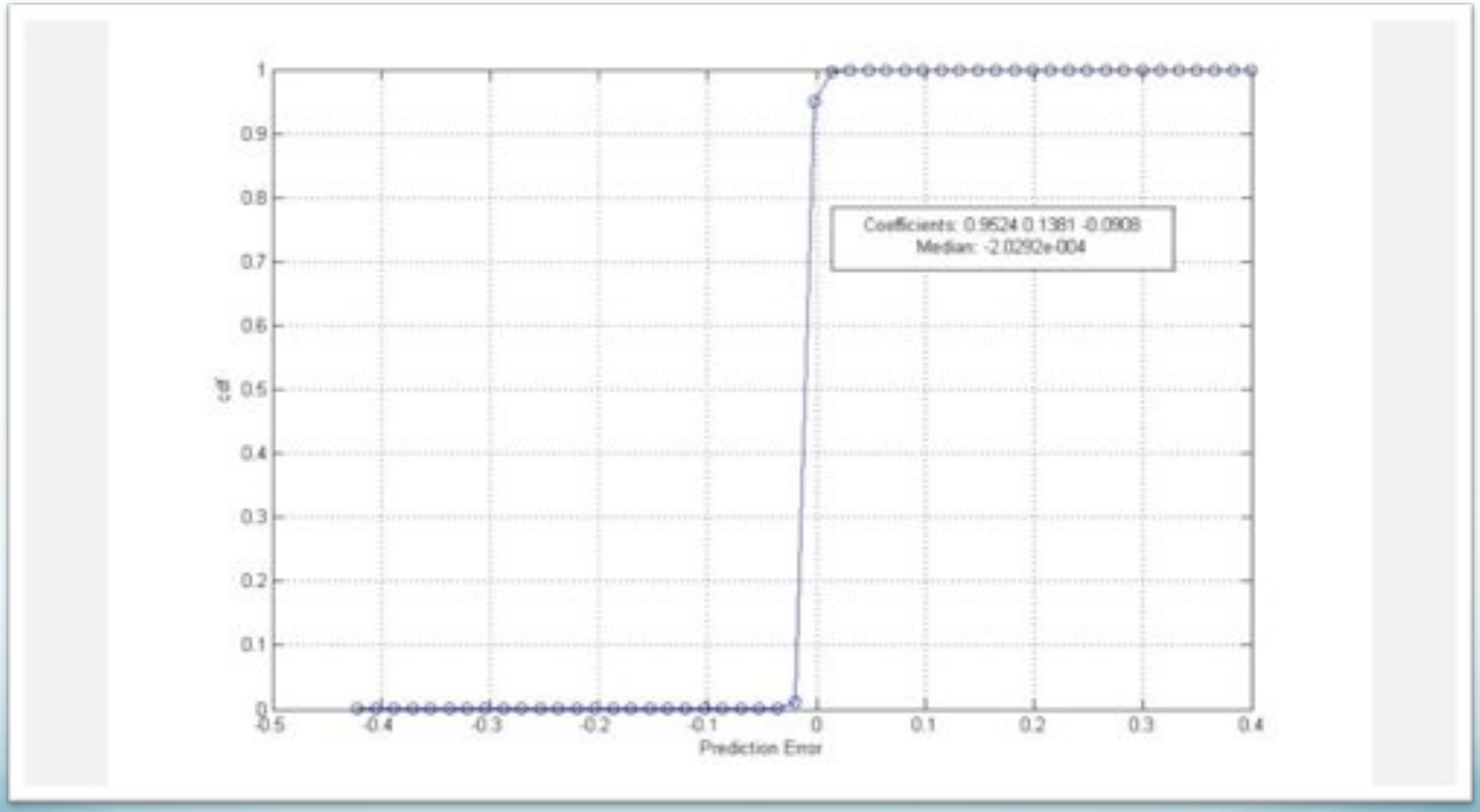
A simple model for CPU utilization

- Utilization: fraction of computing resources actually used, as reported by ganglia, aggegated over the cluster
- Depends on
  - request process, ie usage
  - site scheduling policy. In our case,
    - work-conserving
    - time sharing limited to very light tasks
  - Thus to some extent « intrinsic »
- An AR (3) model is sufficient!

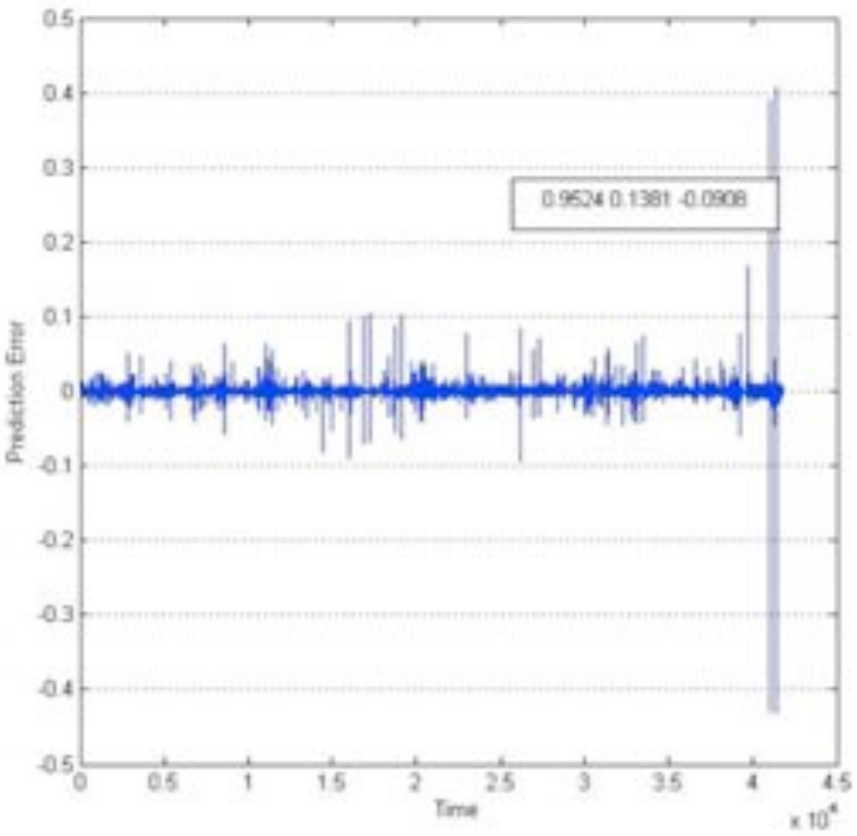
$$X_t = a_1 X_{t-1} + a_2 X_{t-2} + a_3 X_{t-3} + \varepsilon_t$$

- Thus very predictible

Validation of the AR(3): residuals

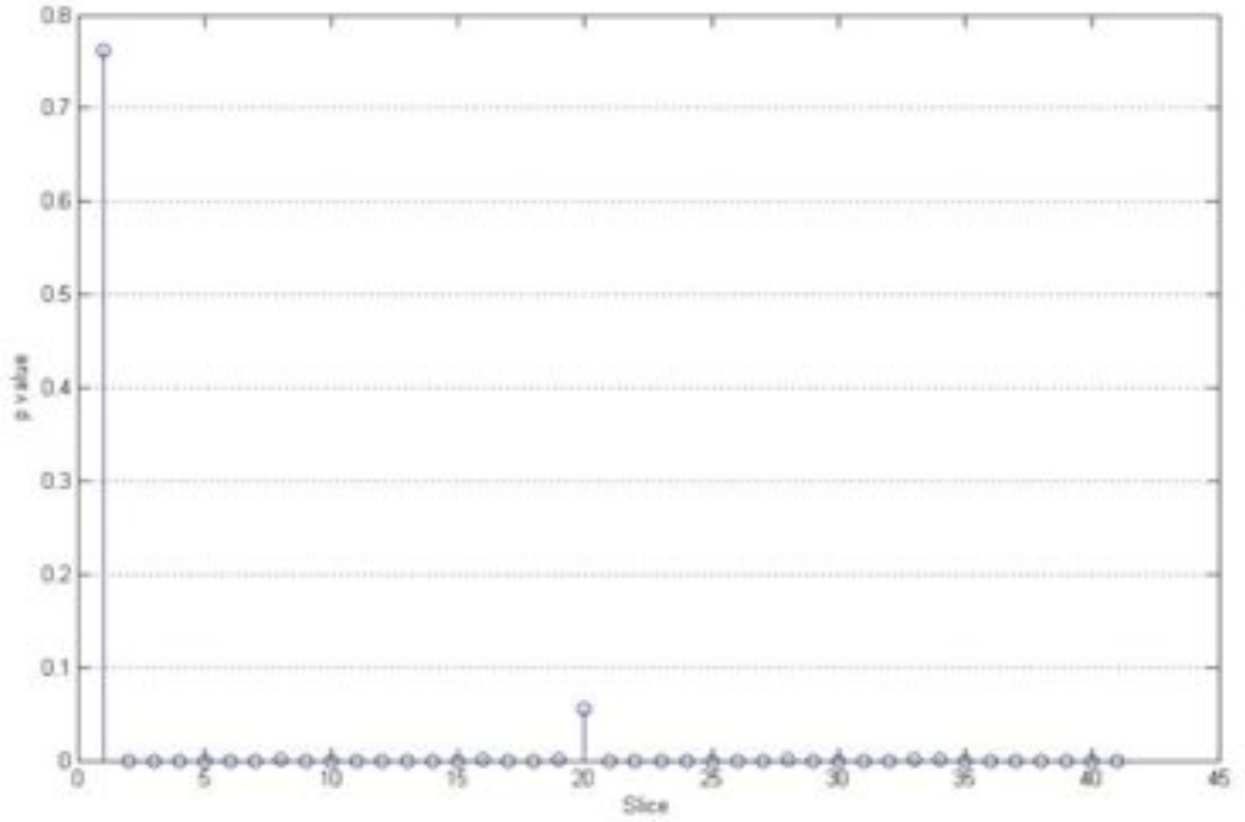


Validation of the AR(3): residuals



Validation of the AR(3): Ljung-Box test of independence of residuals

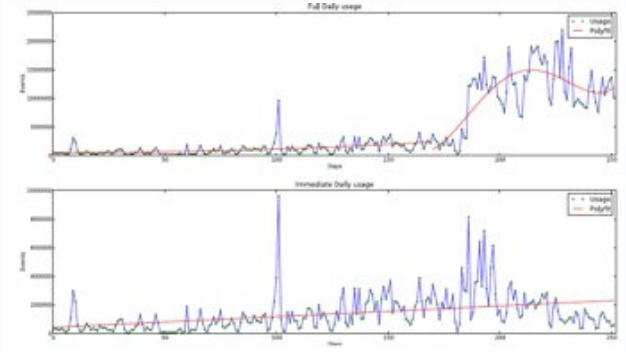
- Portemanteau test of autocorrelation at all lags. Smaller is better
- Global: p=0



Stationarity?

The "physical" process is not stationary

- Trends: Rogers's curve of adoption
- Technology innovations
- Real-world events
  - Experimental discoveries
  - Slashdotted accesses



But AR models imply stationarity?

General idea: piecewise simple models, separated by breakpoints

- experiments give consistent results, and the analysis can be made scalable, but a posteriori.
- online: robust tests do exist

[Towards non-stationary Grid models. Journal of Grid Computing, 9:4]

[Scalable structural break detection. Applied Soft Computing 12:11]

A simple model for energy usage

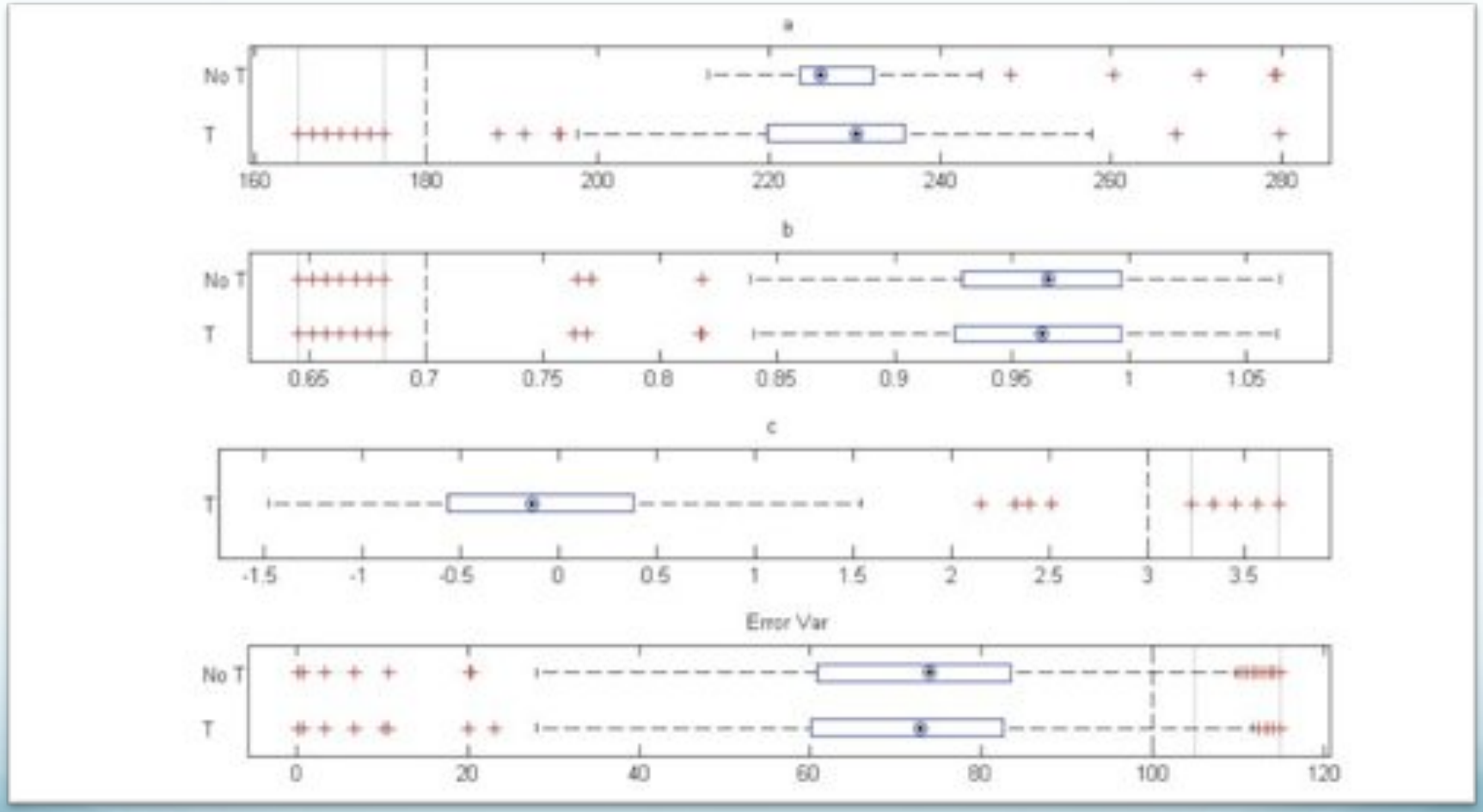
Grid Observatory

- At our coarse sampling frequency, by far the major factor is the CPU load L, and a minor one is the motherboard temperature T. All other ones (memory, network, fans) have no noticeable impact
- A linear model is adequate

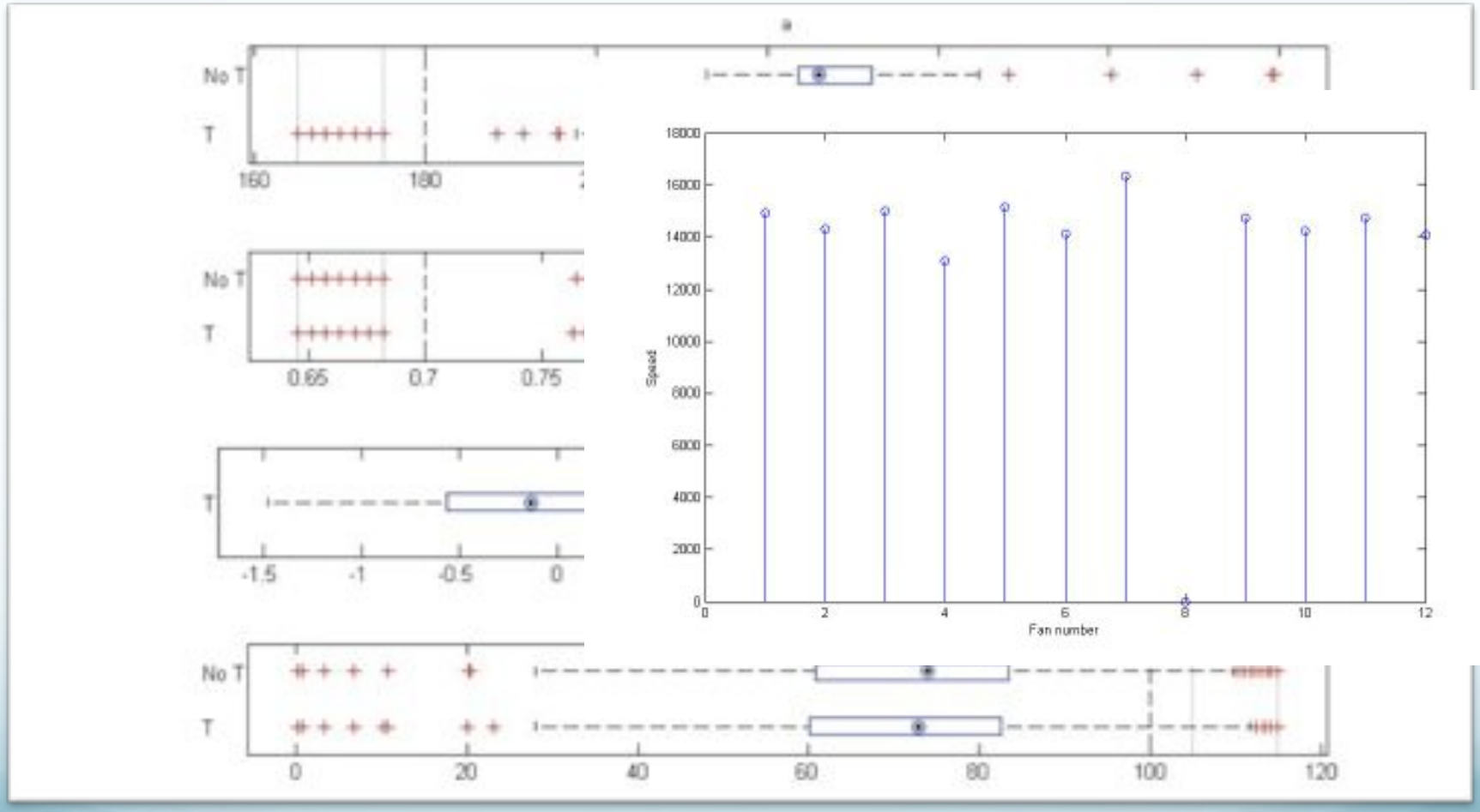
$$P = a + bL + cT$$

- Caveat: CPU only, disks and network components not monitored, part of global (room) energy usage

A simple model for energy usage



Outliers happen!



Opportunities for energy saving

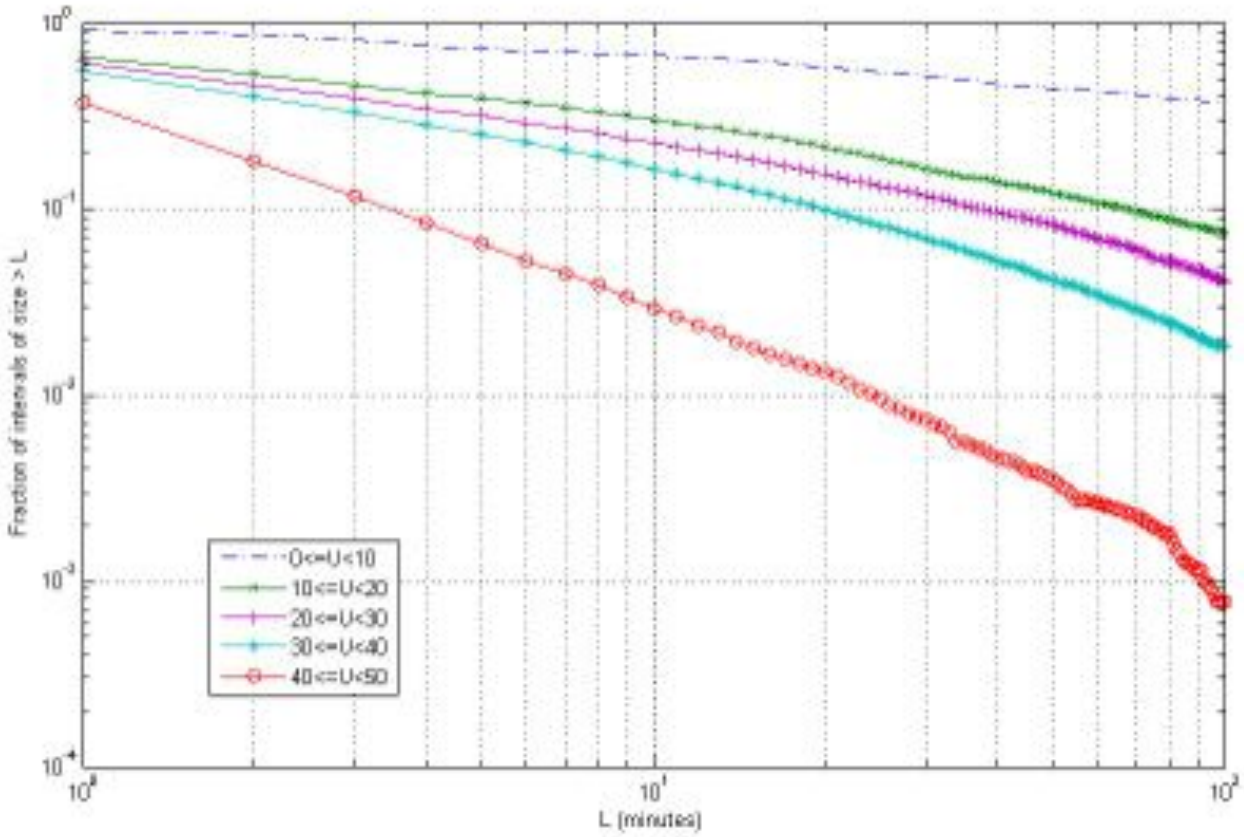
- On/off: moving load and switching off machine, or exploiting energy-proportional IC designs

  eg how profitable would it be to pack work on 60% loaded machines given migration costs?
- Key concept: activity metric(Barroso 2011)

$$A(L, U) = \mathrm{P}\,(l = L\ ,\ u{=}U)$$

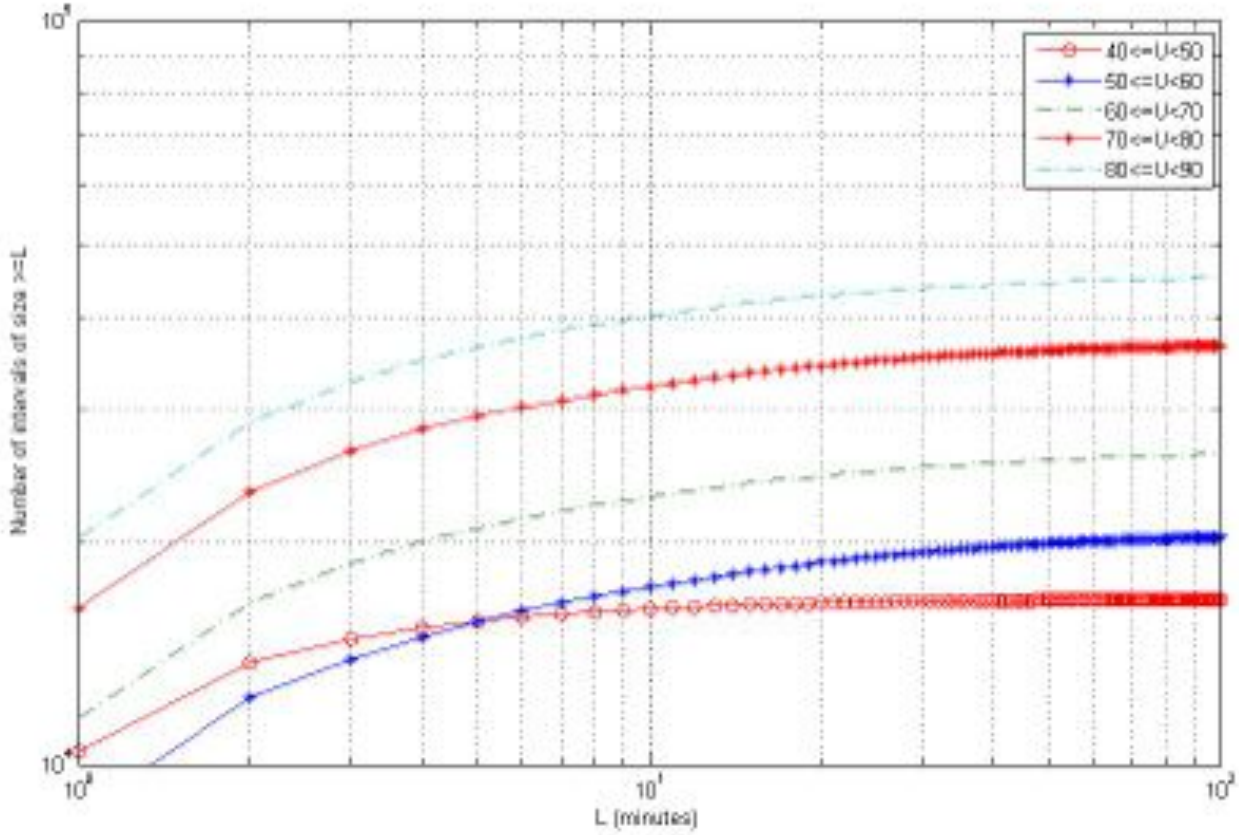
The fraction of time a component spends at utilization U for a time period of L

Activity metric



In one case in ten, the present load at 30-40% will extend to more than 10 minutes

Highly scheduler dependent



Conclusion

The seven pillars of wisdom@ wadi Rum

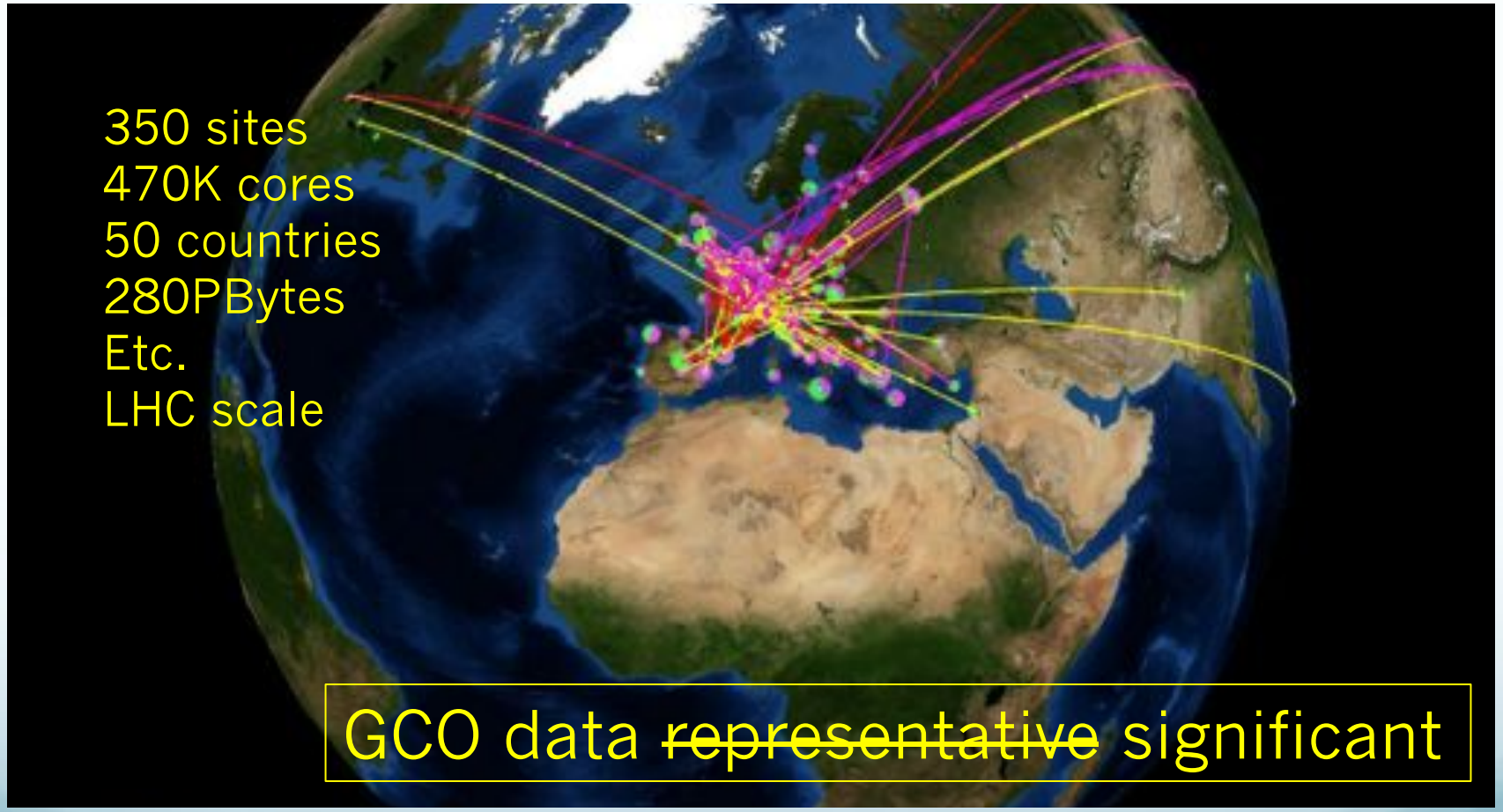
Conclusion

- Automatic acquisition is up, running, scalable and sustainable
  - Robust ontology and data schema
  - But not perfect – as always the case with Big Data
  - And we did not exploit other GO data – specifically individual job duration

Simple linear models experimentally validated

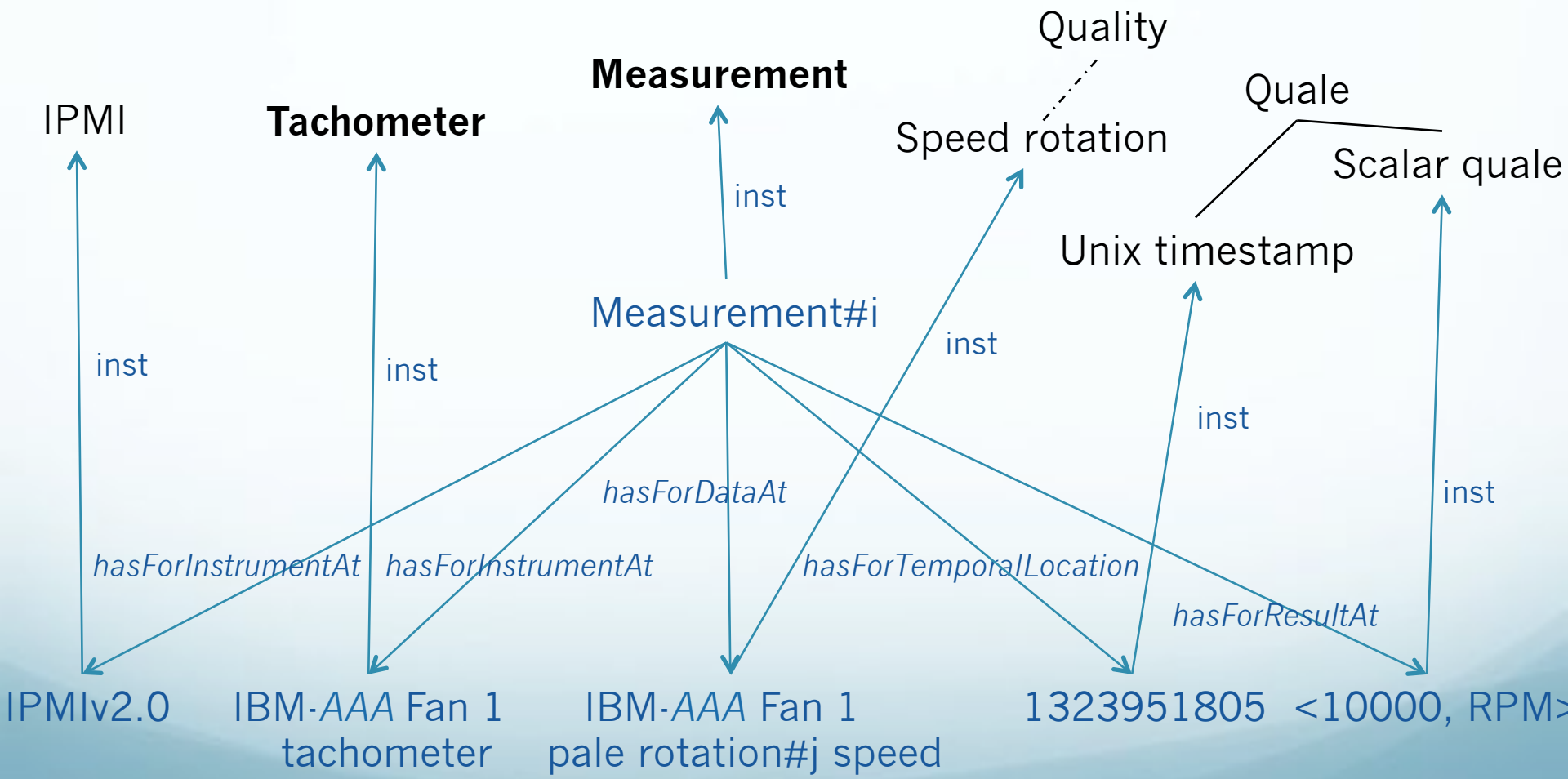
- Real world exploitation would have to integrate breakpoint detection

The full EGI infrastructure
Very large non-profit distributed system



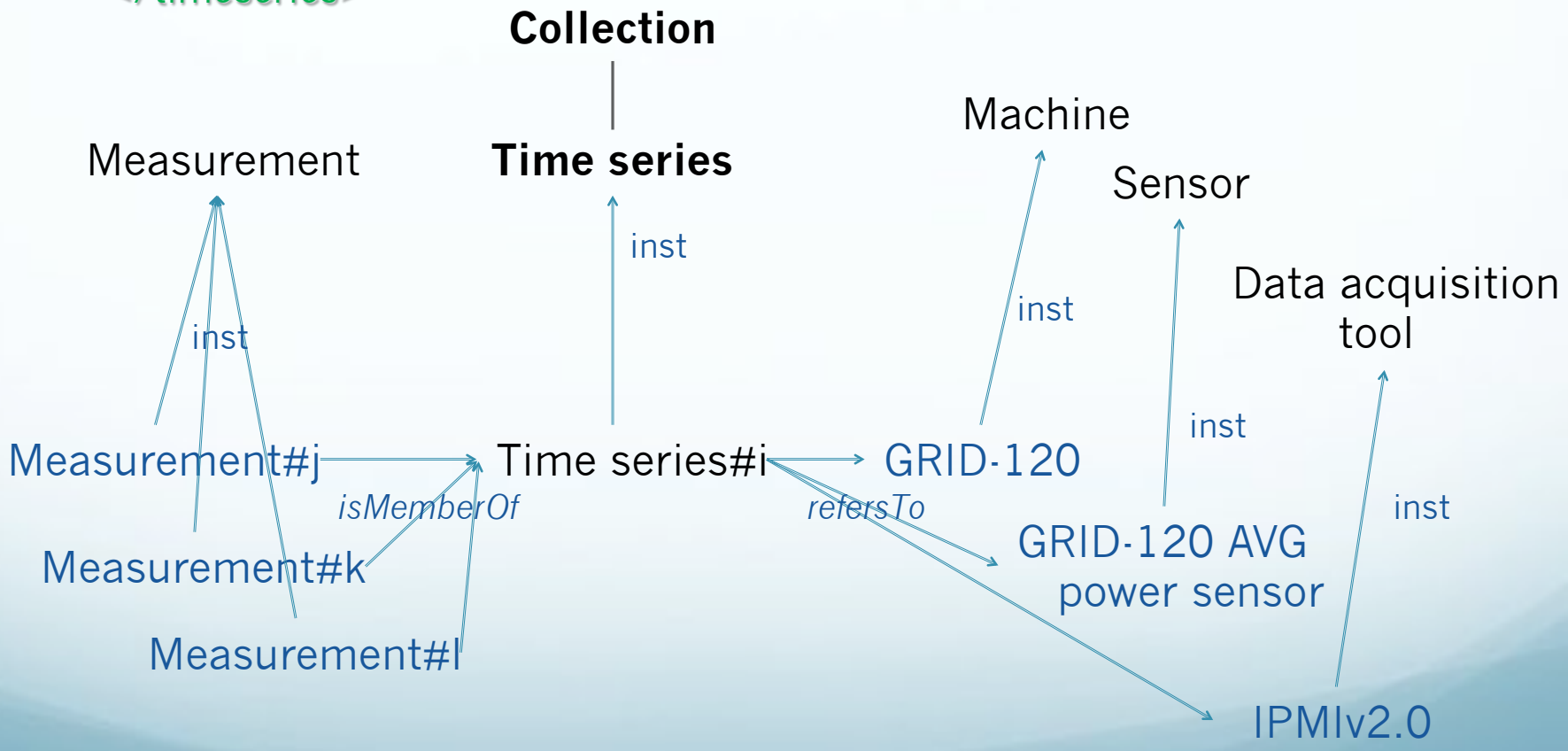
Acquisitions

("IPMI", "FAN1 TACH", 1323951805, 10000)

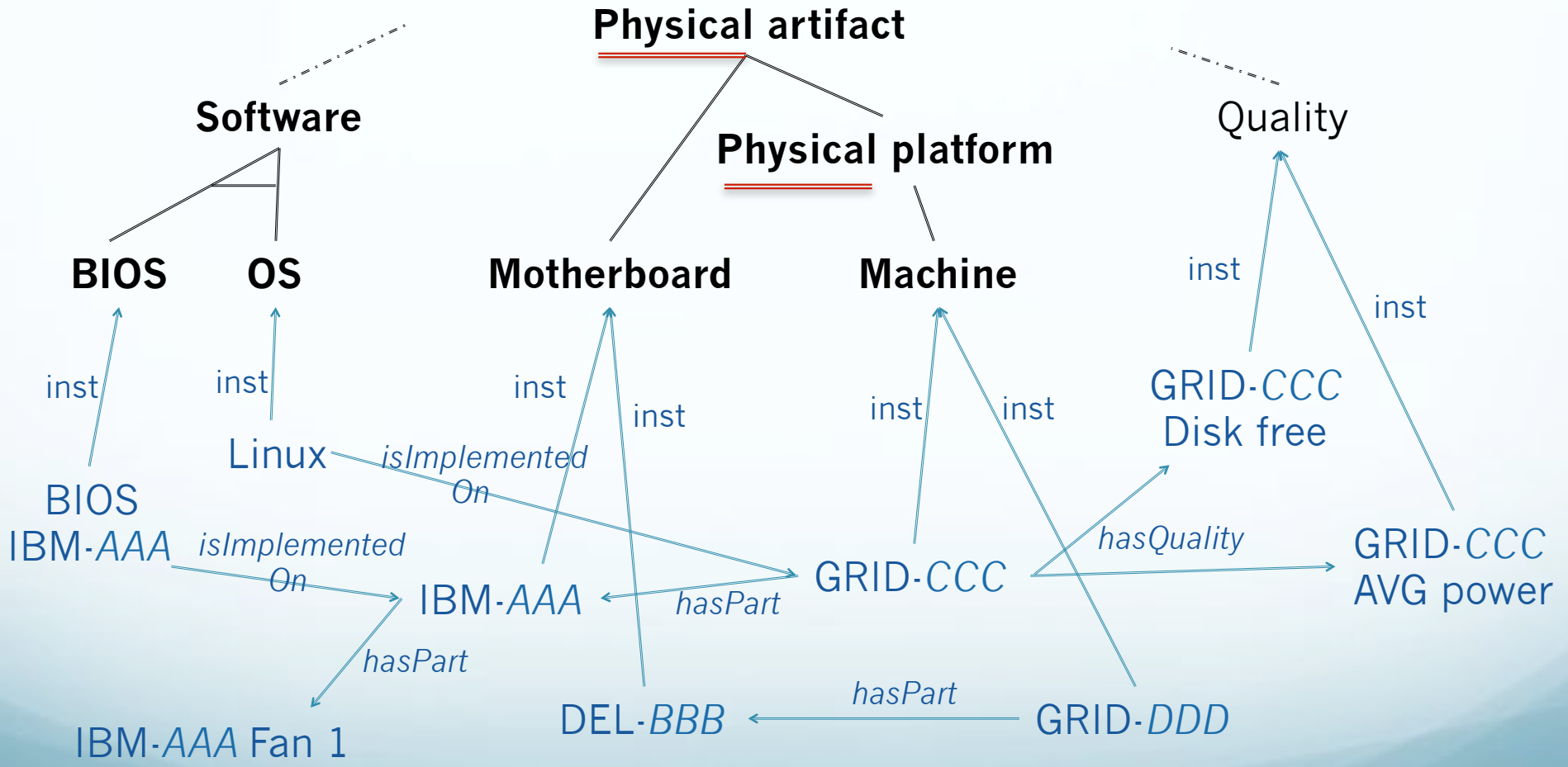


Time series

```
<timeseries machineID="1" machineName="grid120" instrumentID="2">
    <a t="1325942960" v="320.00" />
    <a t="1325943020" v="310.00" />
    <a t="1325943080" v="320.00" />
</timeseries>
```



Motherboards and Machines

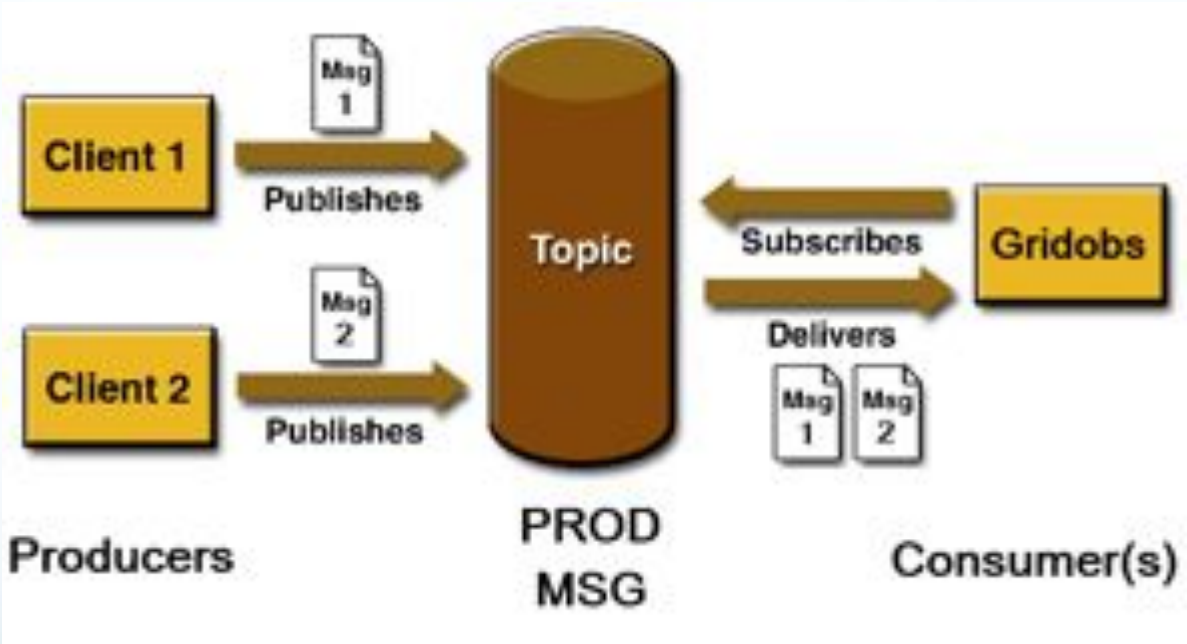


2.1 ActiveMQ for the GCO as a transportation layer

- Previously acquiring data with a « PULL » method
  - Connect to each maching
  - run acquisitions
  - Retrieve values
- Need access to each machine
  - IPMI access
  - Proper network routing (LAN only – not for distant sites)
- Need to run 250+ parallel queries
  - Concurrency problems, disk overload

2.1 ActiveMQ for the GCO as a transportation layer

- Now using EGI PROD MSG



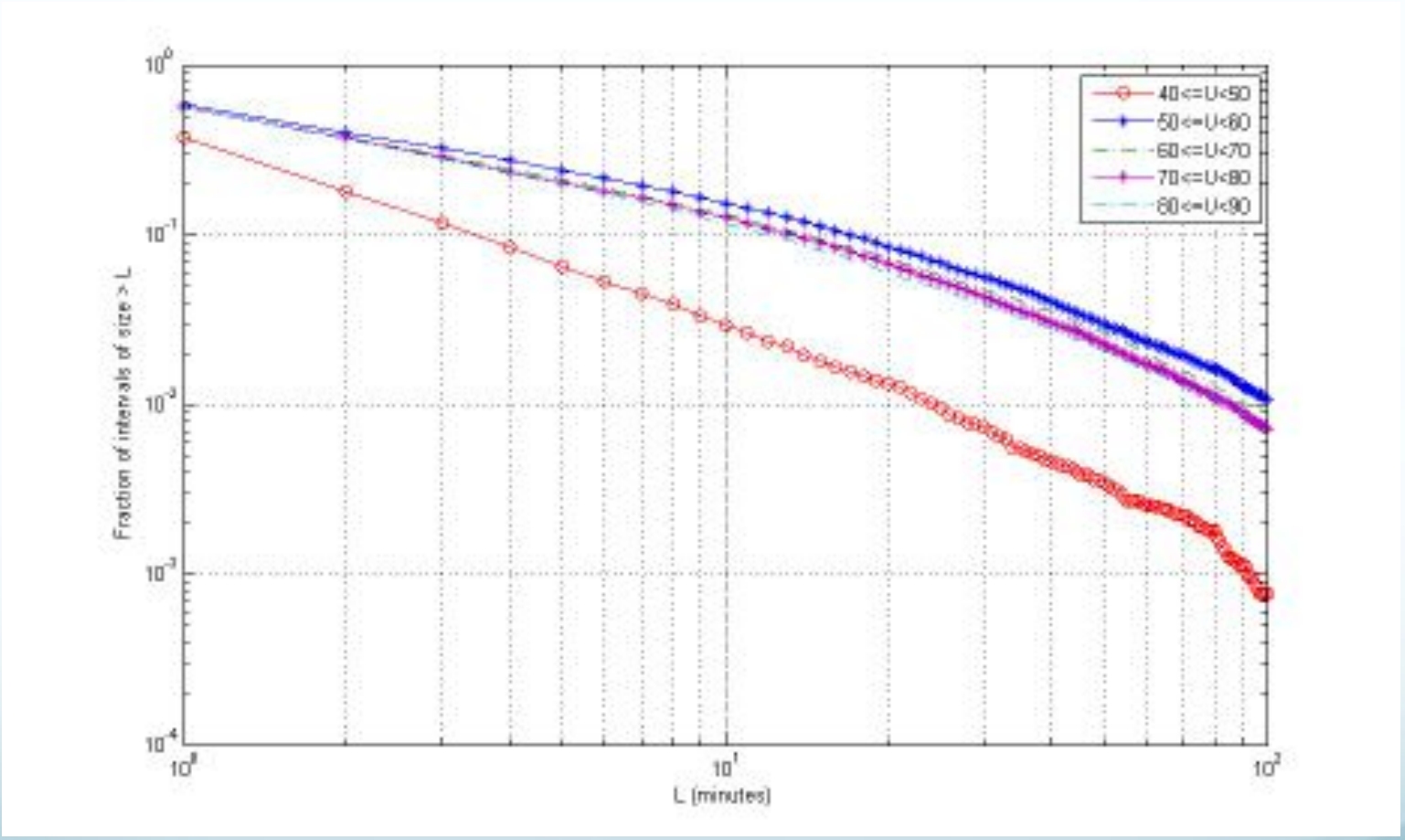
2.1 ActiveMQ for the GCO Pros

- Fault tolerant
  - Stores messages when no consumer is available
- Highly scalable
  - Producers and consumers can be transparently added
- Generic in many ways
  - Can be used with different programming languages
  - Can be reused for other applications, e.g. L&B

2.1 ActiveMQ for the GCO Cons

- Requires an external server
  - Using EGI's PROD MSG servers
- Producers run on middleware => can no longer monitor hardware directly
- A microprotocol has to be developped to describe the transported data

Activity metric – no simple model



Counterintuitive: opportunities seem to be better at higher load

Fractions are misleading