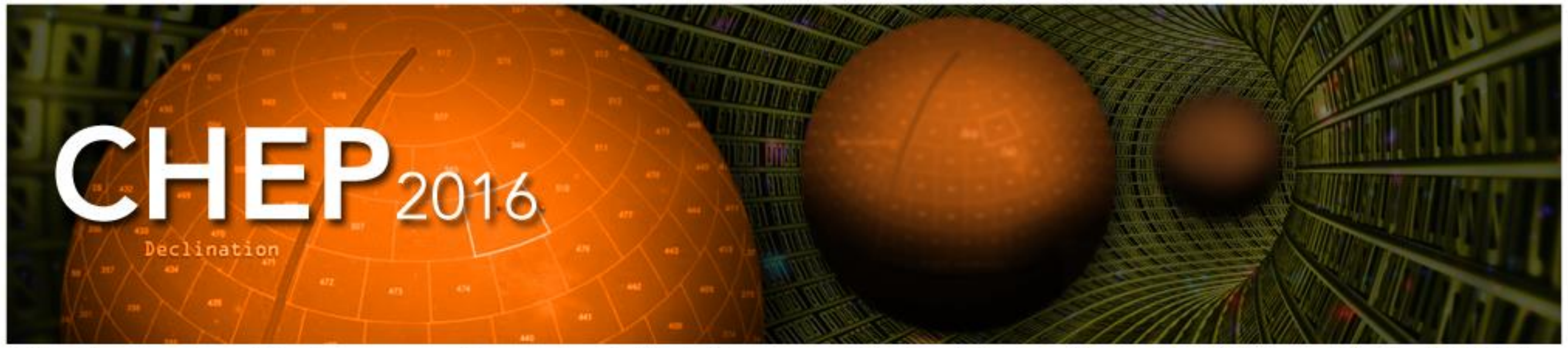


22nd International Conference on Computing in High Energy and Nuclear Physics, Hosted by SLAC and LBNL, Fall 2016

First results from a combined analysis of CERN computing infrastructure metrics

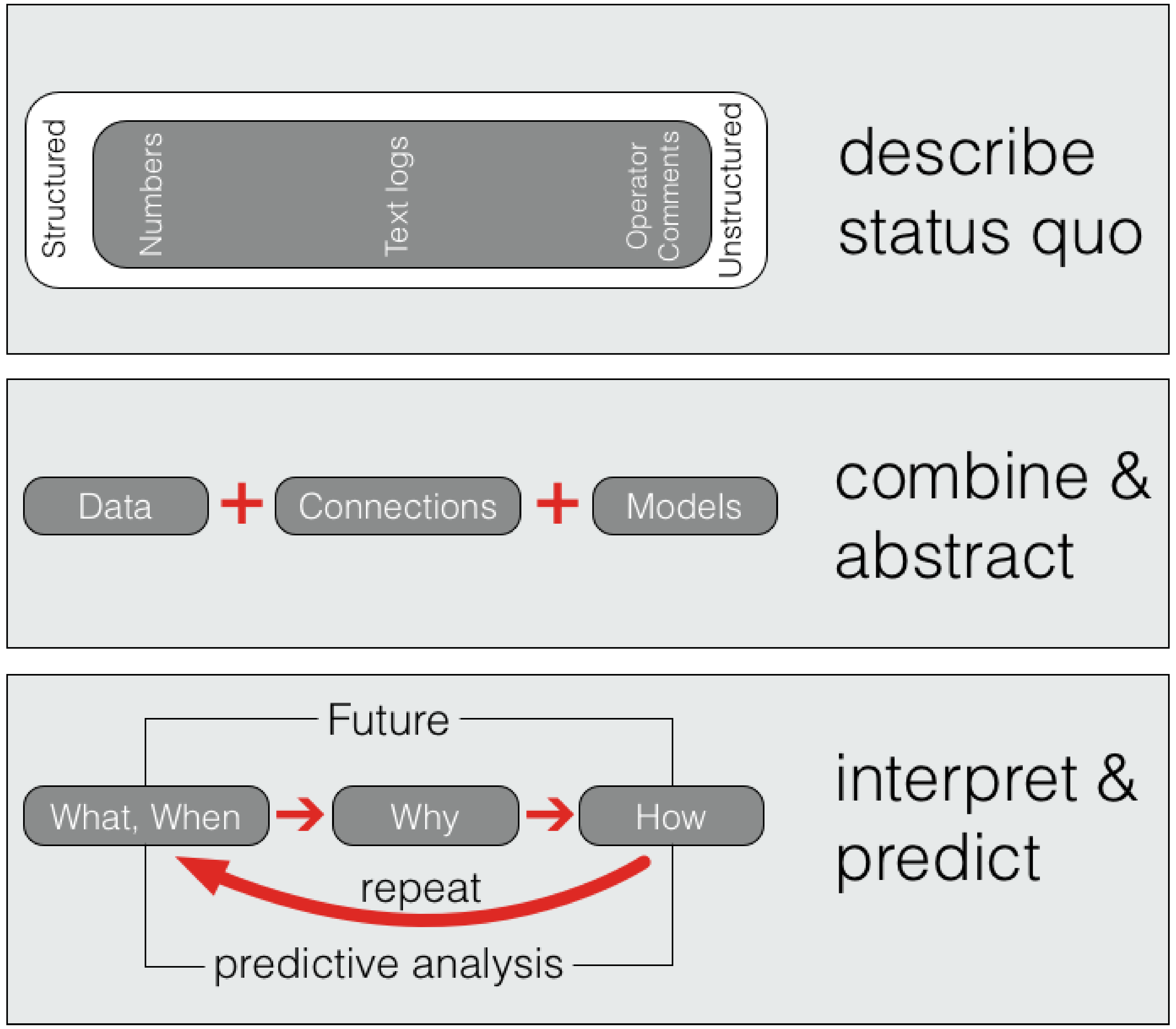
D. Duellmann - CERN, Switzerland
C. Nieke - Tech. Univ. Braunschweig, Germany

presented by Oliver Keeble

Analysis Working Group Motivation

- Goal: **quantitative** understanding of computing infrastructure
    - measure service usage and their evolution
    - To optimise science throughput / CHF
- End-to-end: using metrics from user and across layered services
    - Eg within large (many jobs), well-identified physics tasks
        - optimise aggregate events/s delivered to experiments

Three Main Phases



Structured

Numbers

Text logs

Operator Comments

Unstructured

describe status quo

Data + Connections + Models

combine & abstract

Future

What, When → Why → How

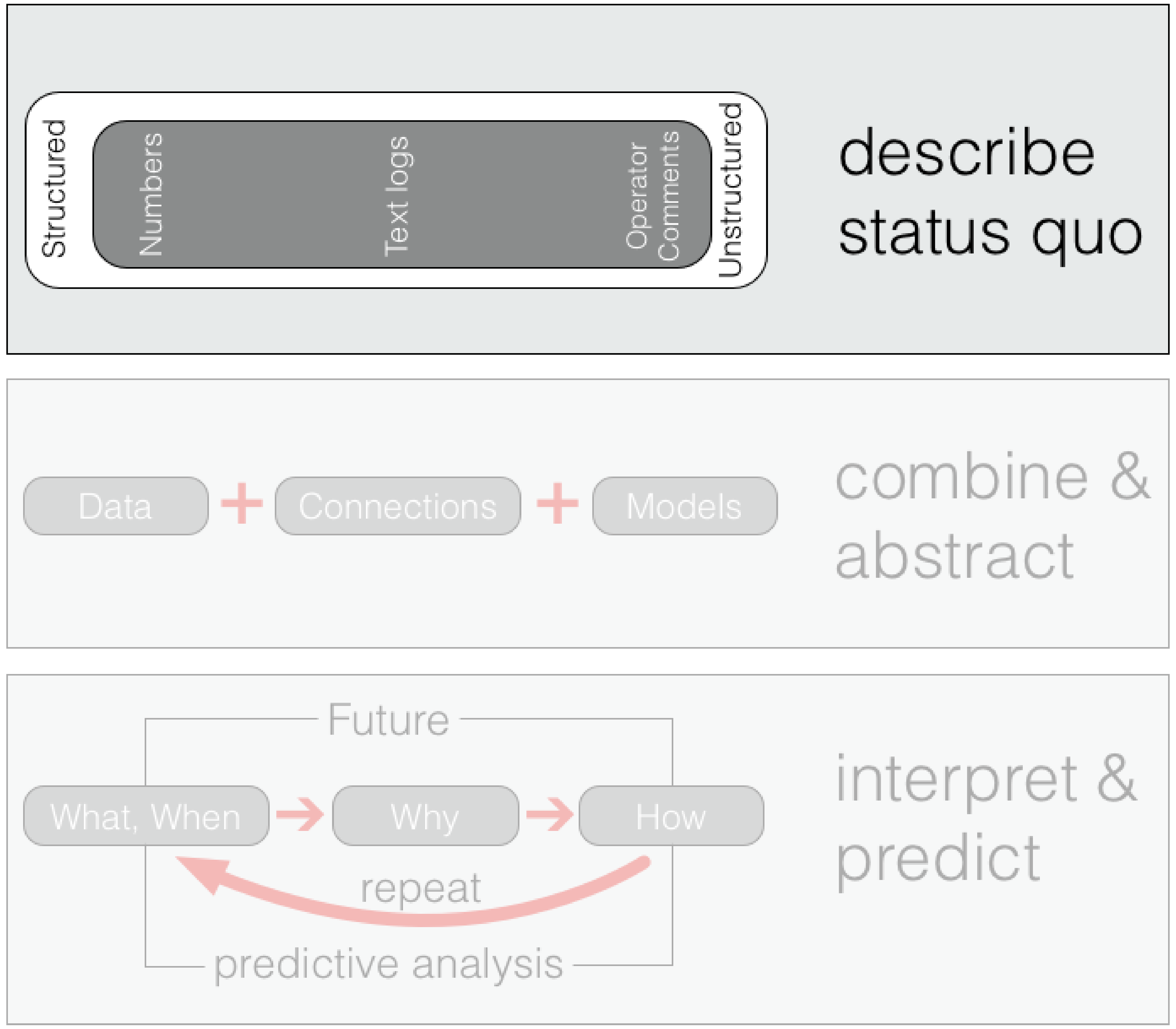
repeat

predictive analysis

interpret & predict

- Tools
  - statistical analysis {machine learning}
- Follow rational flow
  - extract relevant statistics from raw metrics and logs
  - connect trusted data into models
  - check model prediction and improve

Metric Collection



Structured

Numbers

Text logs

Operator Comments

Unstructured

describe status quo

Data + Connections + Models

combine & abstract

Future

What, When → Why → How

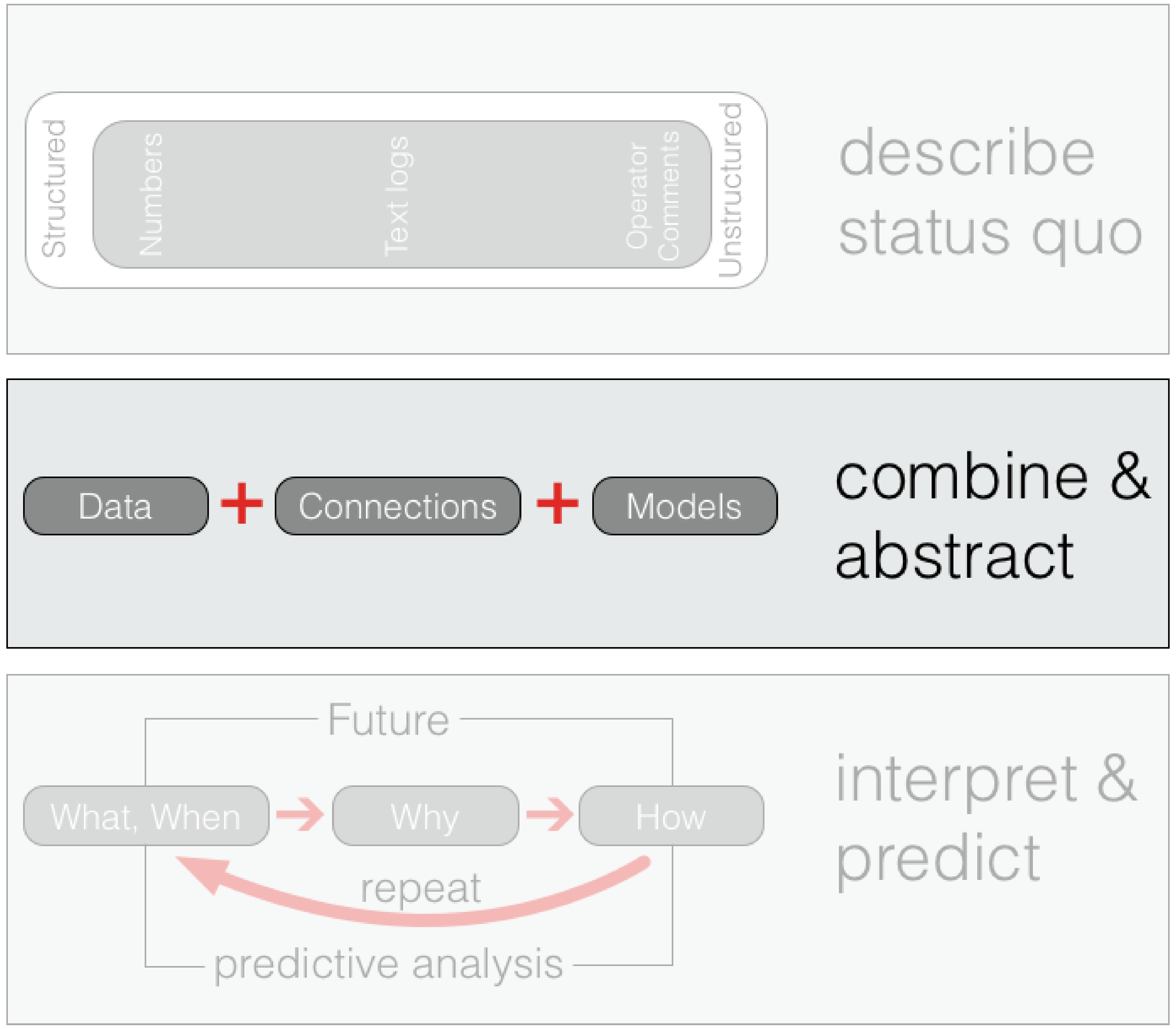
repeat

predictive analysis

interpret & predict

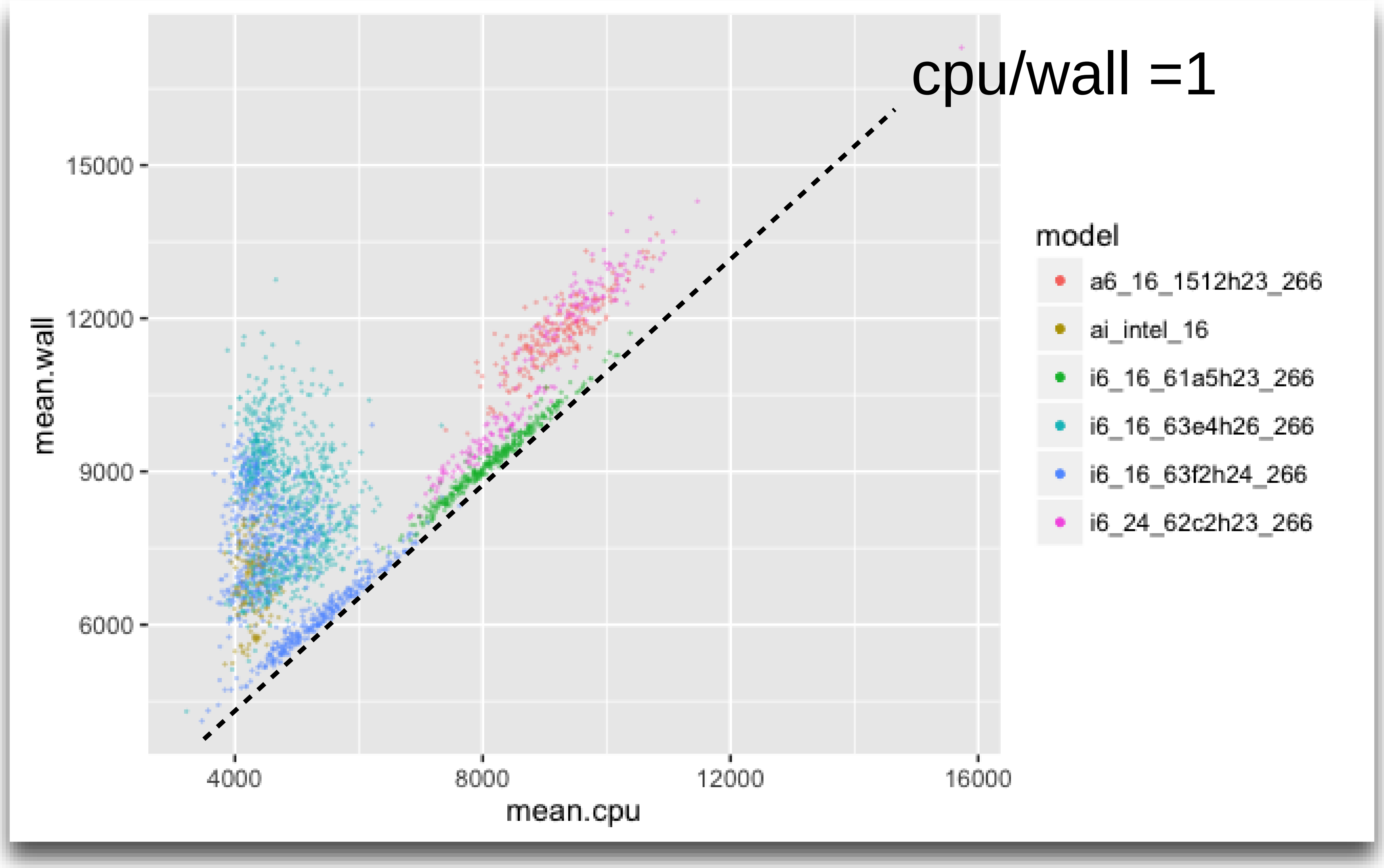
- **Collection** via *IT monitoring project*
- select and summarise **relevant metrics**
  - Find & remove unexpected / unintended access patterns
- **To what level** can we trust our metrics & assumptions?
  - Evaluate data quality: eg accuracy, units(!)
    - data that has not been used quantitatively yet has likely problems
- Simple **quantitative cross-checks**:
  - eg for CPU
    - $\sum \text{job}_{cpu} \sim \sum \text{sched}_{cpu} \sim \sum \text{host}_{cpu}$ (any significant losses?)
  - eg for disk
    - $\sum$ disk I/O ~ $\sum$ user I/O + $\sum$ internal I/O (ratio expected?)

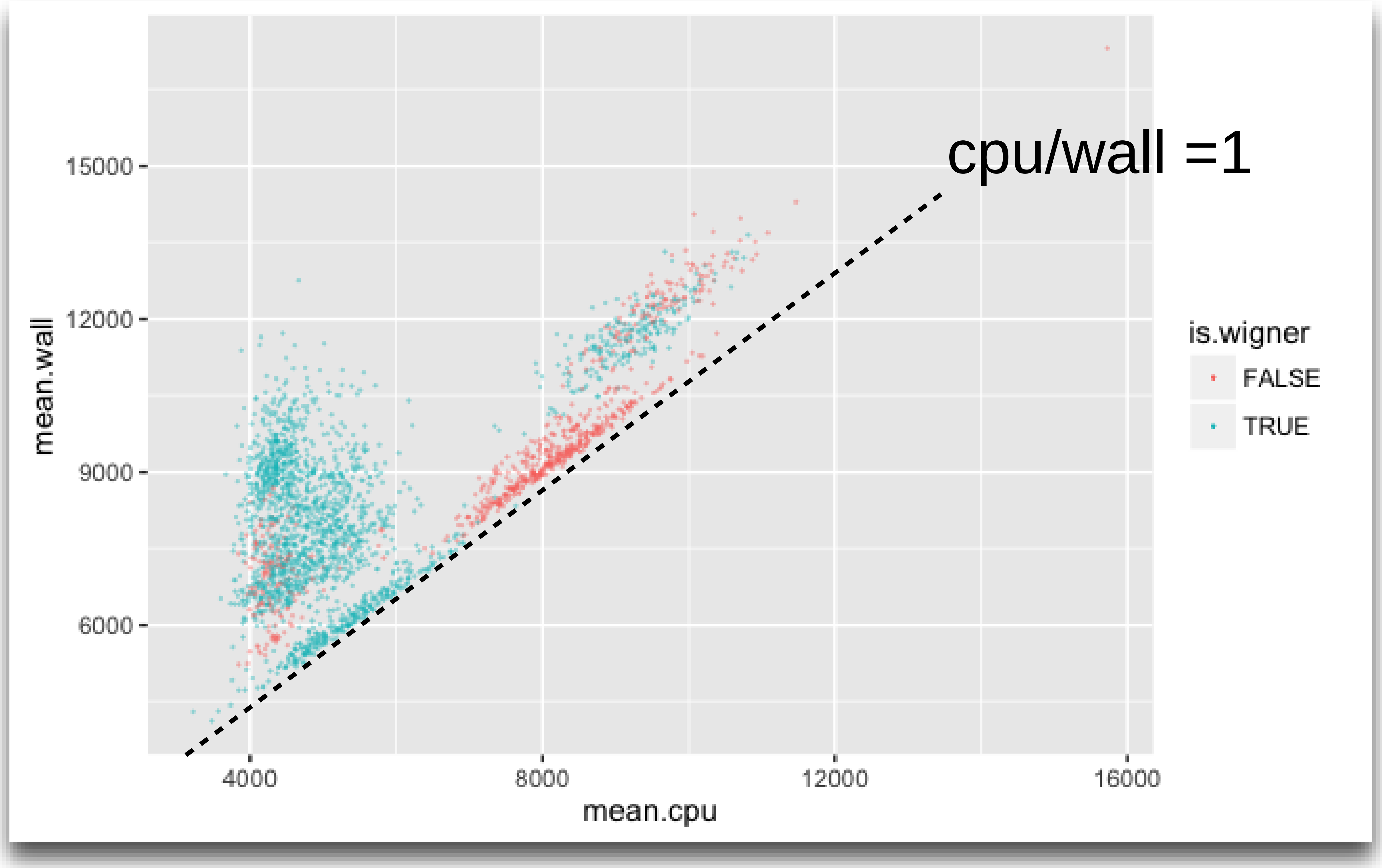
Connecting Data



- Involved in several experiment performance studies
  - Starting point: why do users/service providers see:
    - **slow** file **access**? **inefficient CPU** usage?
    - differences: Wigner vs CERN, CERN vs T1, etc..
  - **where is** the **bottleneck? where should be?**
- Connected data from experiment, storage, batch
  - connected infrastructure data: LAN db, hardware db
  - enables correlation with location, hw type, HEPSPEC

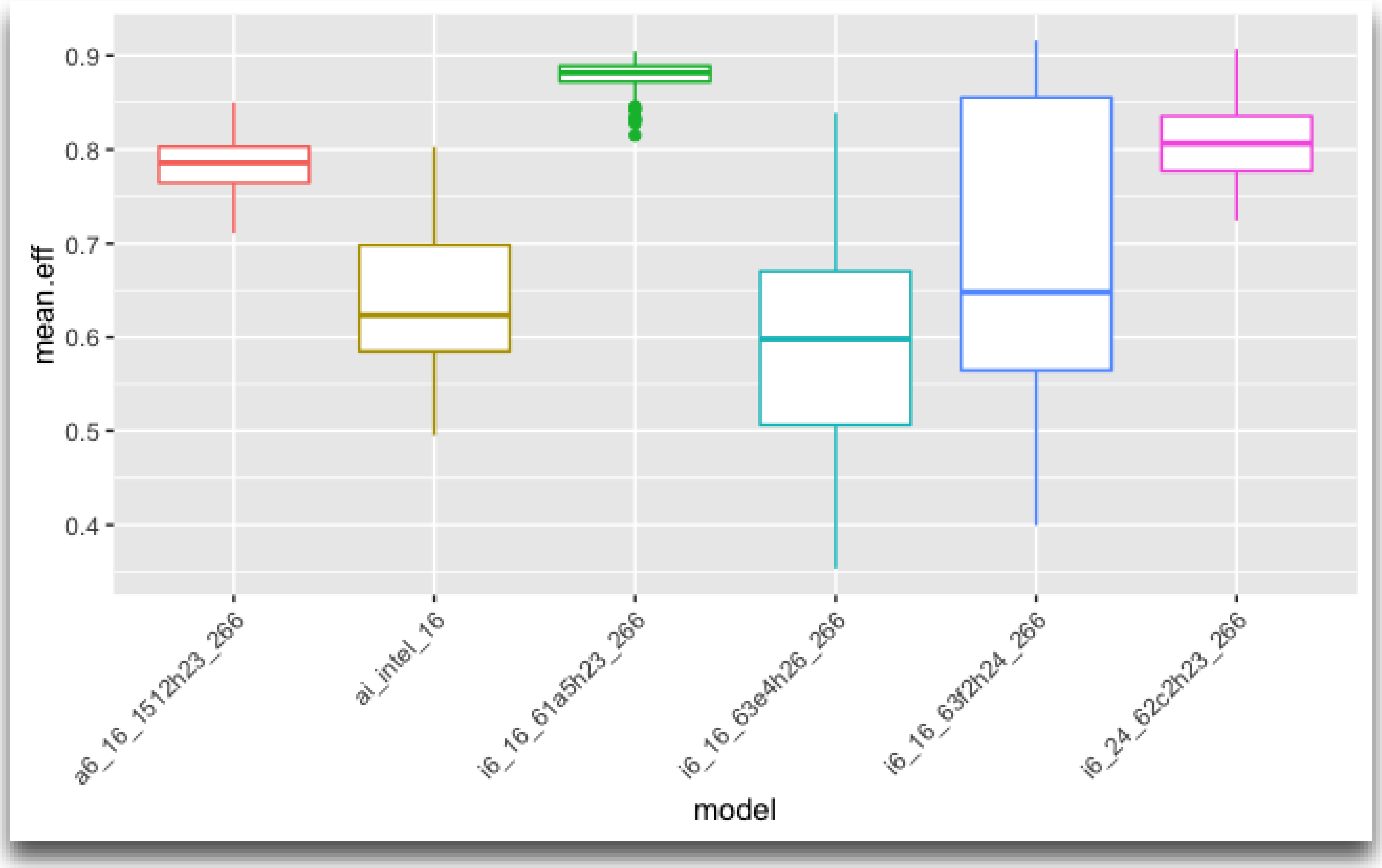
A few simple examples -
machine “efficiency” versus H/W and location



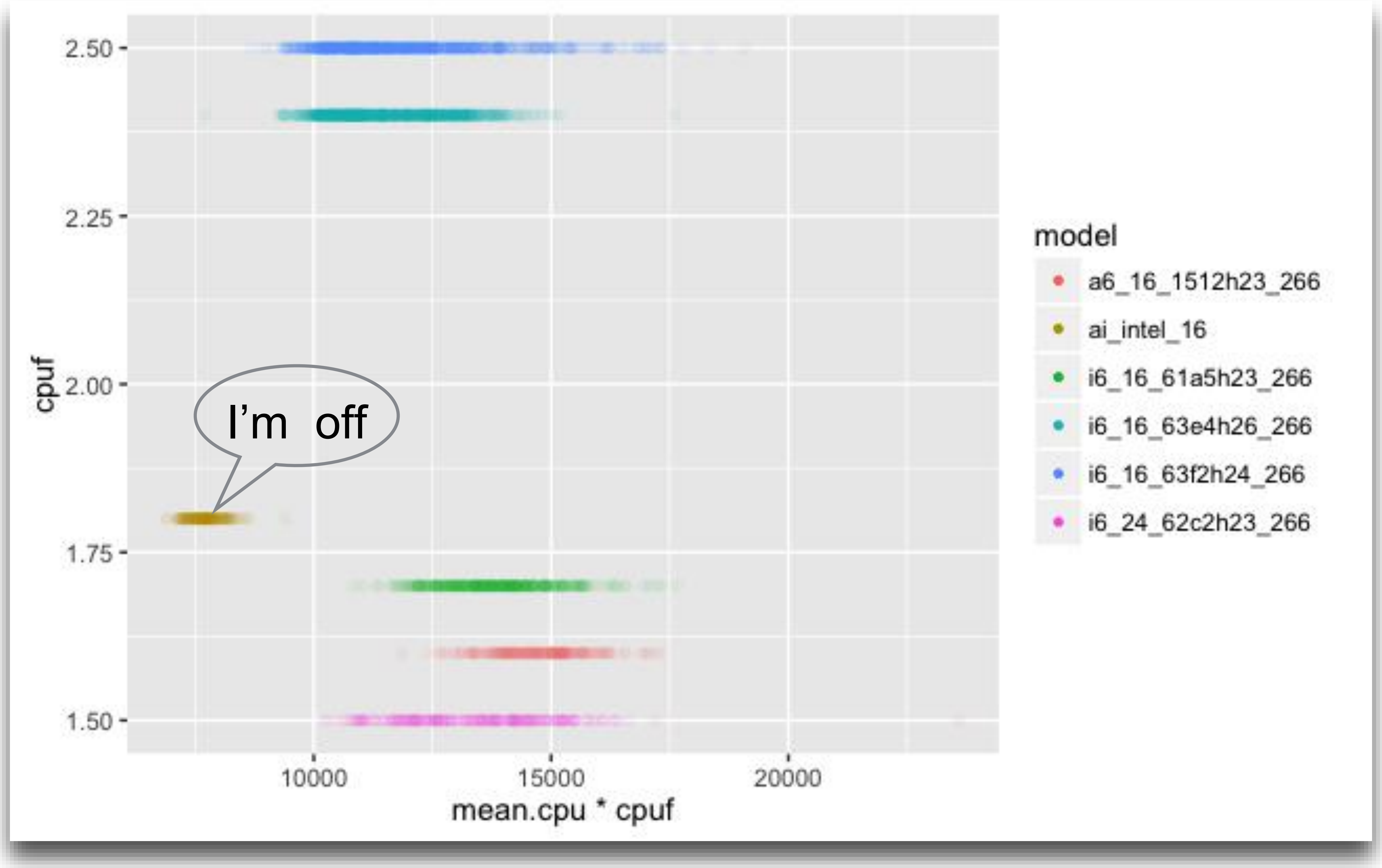


A few simple examples

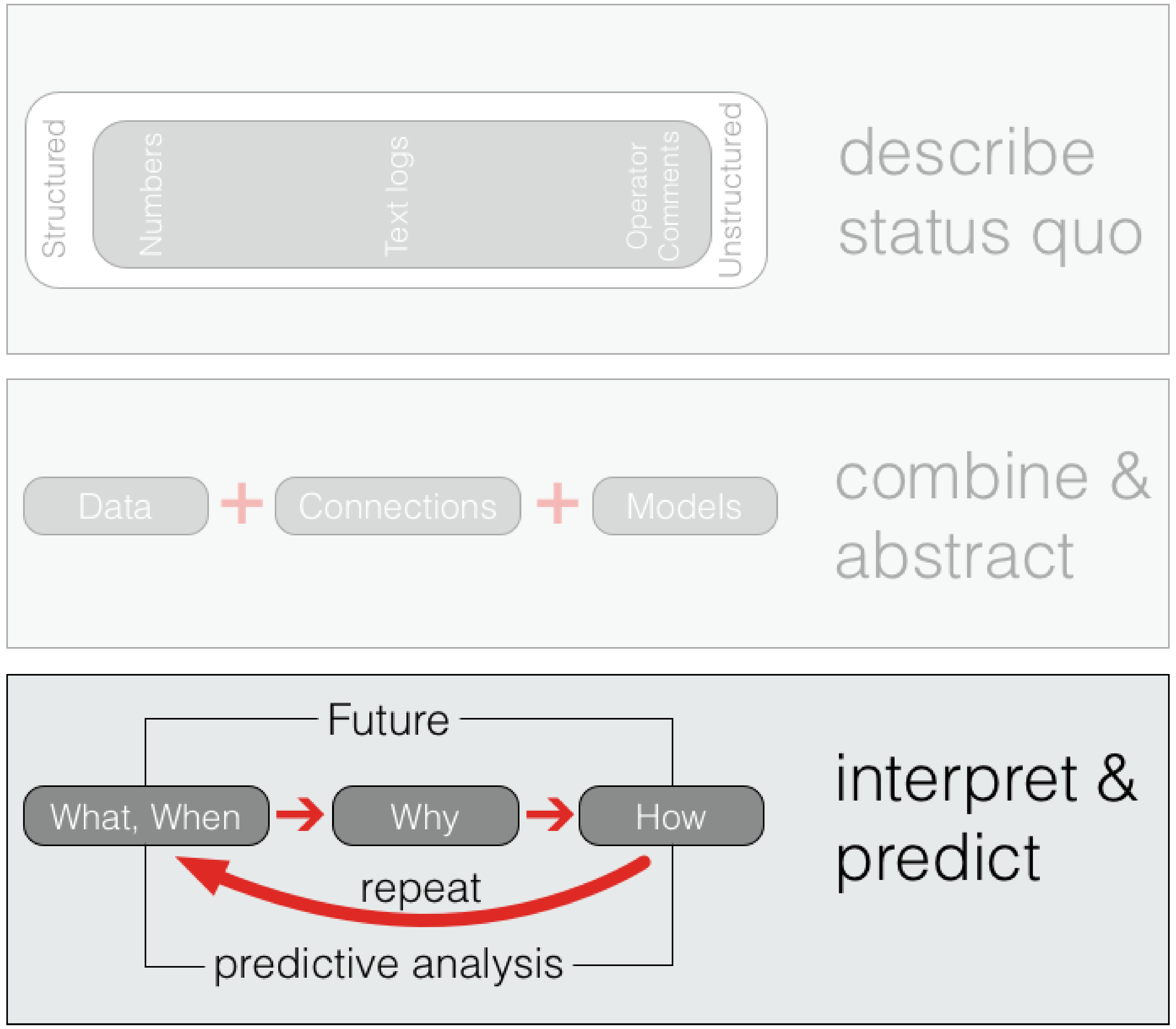
“Efficiency” versus H/W types



CPU Performance Calibration check



Model Predictions



- Evaluate (simple) predictive models: Can we construct a higher performance system for the same price?
- Simplest case: CPU bound
  - **CPU benchmarks** + memory optimisation => MC throughput
- Not CPU-bound case - balance between
  - CPU, WN storage, LAN storage, WAN storage, network

In progress: ML job classifier

- Can we automatically classify jobs to be
  - into: CPU-bound, local I/O-bound, site I/O-bound, “single” file-bound
- Metrics used: experiment (task), lemon (local disk), batch (cpu), EOS (site disk)
  - Evaluating: **simple cut model** and a **random forest** trained with known jobs
  - Classifier output: optimisation hints
    - job placement: e.g. these jobs (don’t) need a local SSD
    - file replication: e.g. these files (don’t) need additional replicas

CPU Benchmarking Motivations

- Compare hardware performance
  - Before purchasing hardware
- Optimising software / configuration
  - Maximise performance for a given workflow mix
- Monitor performance
  - Observe shifts over time
  - Detect outliers / failures / bottlenecks
  - Measure current performance of volatile resources
    - (e.g. cloud VMs)

Current Benchmarking Practice

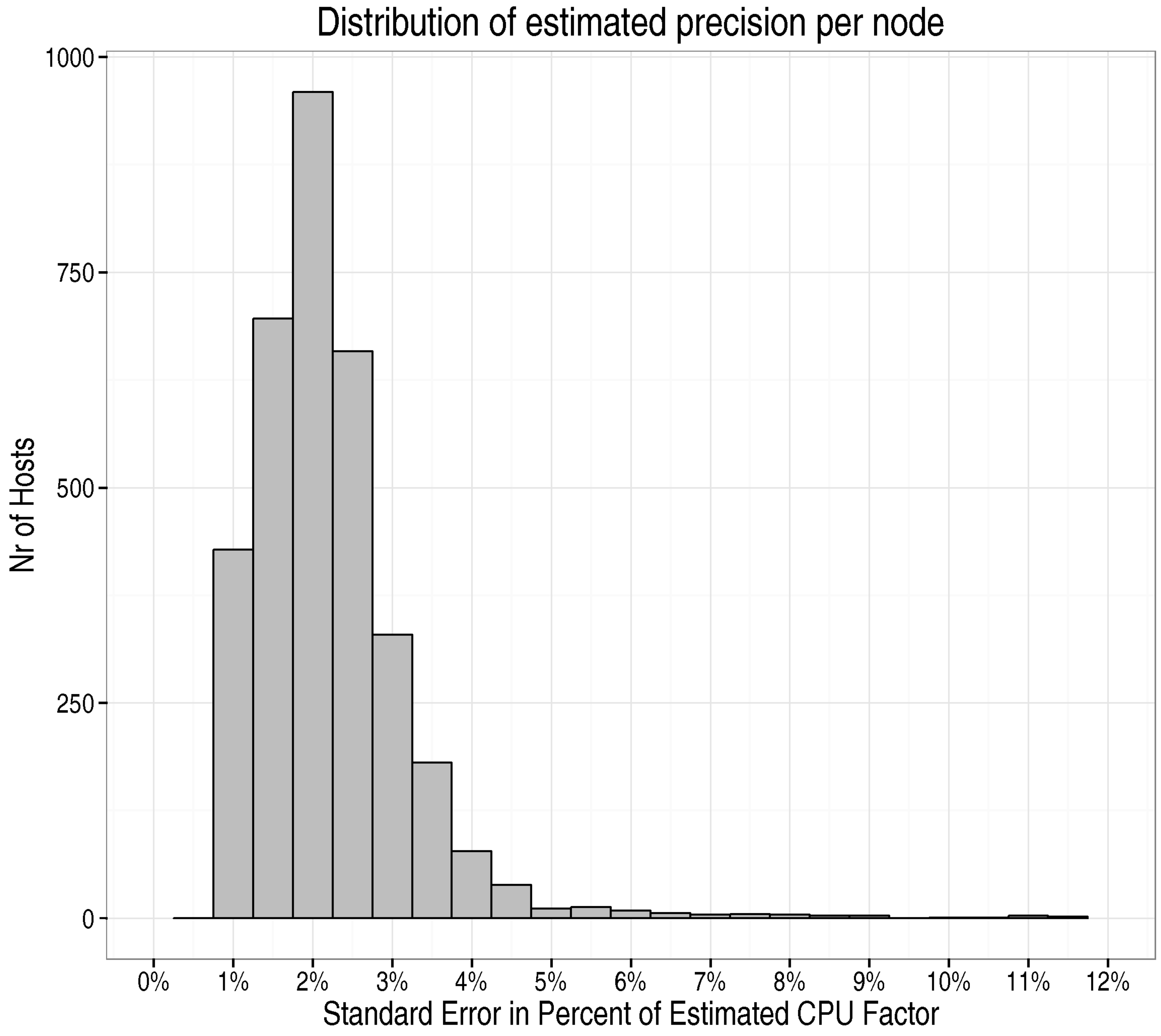
- Long benchmarks (HepSPEC06)
  - High precision (reproducible per machine)
  - Expensive (~8 hours runtime)
  - Intrusive (Node has to be taken out of production)
  - Suited for initial hardware tests

- Short benchmarks
  - Run before each job (e.g. DIRAC, KV, fastBmk etc.)
  - Lower intrusion (10-15 min benchmark vs. ~hours jobs)
  - Suited for hardware monitoring

- But do they represent the actual workflows?
  - HepSPEC designed to represent a mix of HEP workflows
    - But does it?
  - Short benchmarks even more questionable
    - Typically cover only specific areas (e.g. only Monte Carlo)

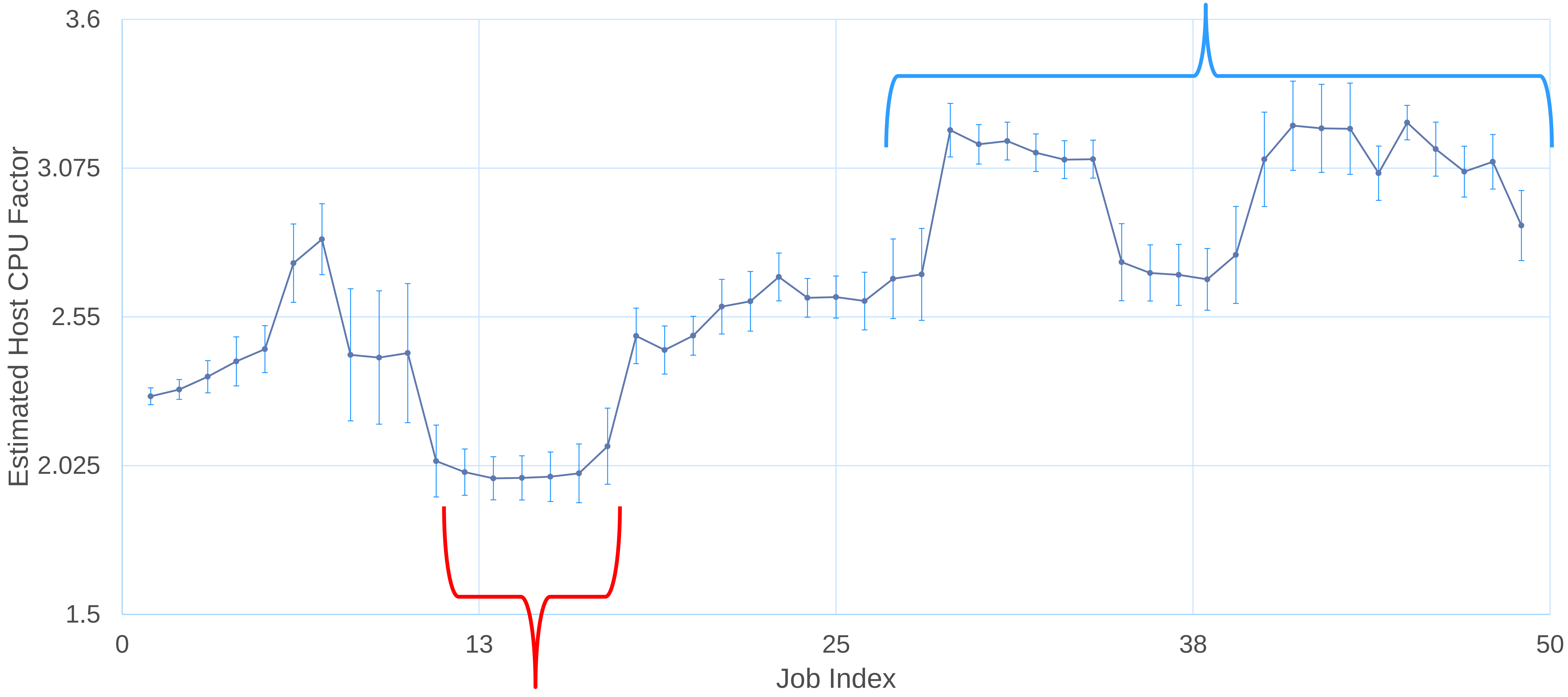
Passive Benchmark

- Basic Idea:
  - Take the workload as set of benchmarks
    - Assume jobs per task are equal, compare runtime
    - Based on existing monitoring logs
- Advantages:
  - Zero intrusion, basically no overhead
  - Always representative (the benchmark **is** the workload)
- Application:
  - Observe performance during operation
  - Compare configurations by performance on the actual workload
- Accuracy / Precision
  - Experiment on LSF dataset: ATLAS and CMS, 3 months
  - Equal or better prediction of performance than HepSPEC06
  - Precision per node is below 5% error for 98% of nodes



“Live” Performance Monitoring

CPU performance computed on sliding window of 4 tasks



sister VM empty

Node failure and reboot

Summary

- Statistical analysis activity of CERN infrastructure performance has started
  - IT Analysis Working Group [meetings] and [twiki]
- Input metrics and analysis environment are now in place and allowed first quantitative studies
  - close collaboration with IT monitoring project and Hadoop service
- Promising new approach to determine CPU performance directly from the workflow
  - sufficient resolution to spot time dependent efficiency changes
- Growing interest from experiments and other sites in statistical and ML optimisation