Servers in the Mist

Operating Systems for Server Utilities

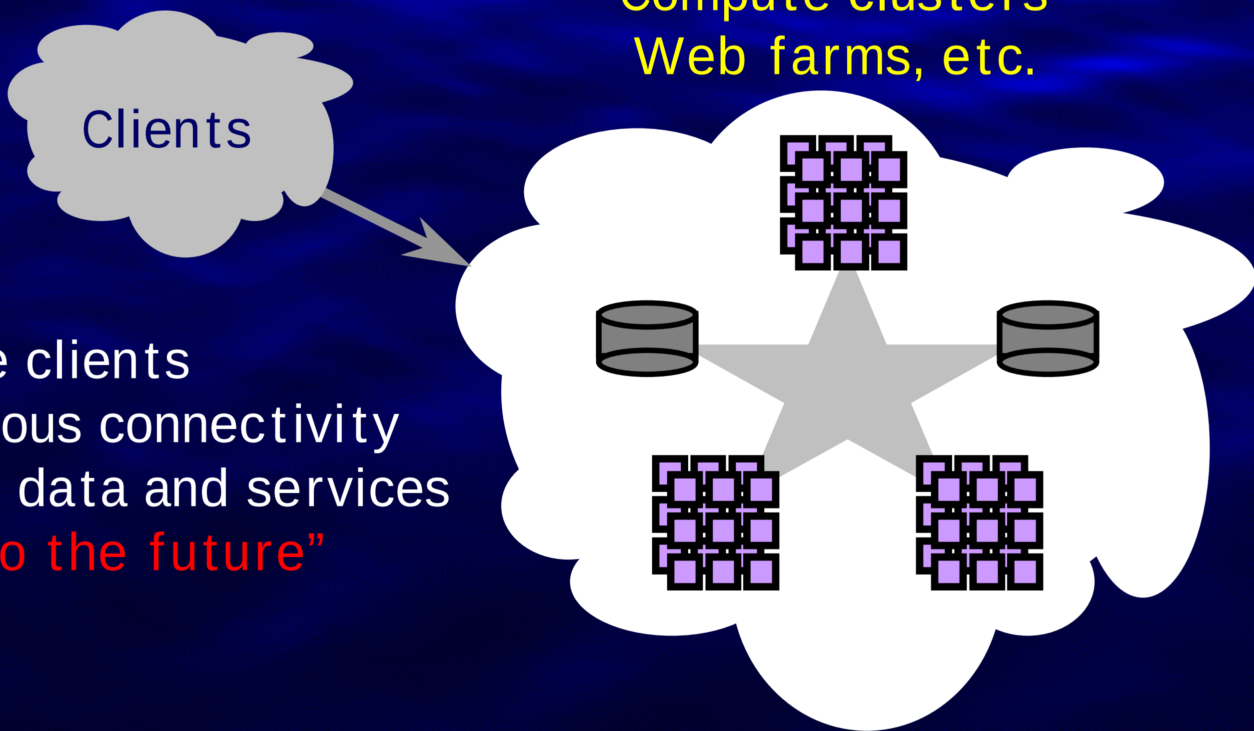
Jeff Chase

Internet Systems and Storage Group
Department of Computer Science
Duke University

The Server Cloud

Diverse clients
Ubiquitous connectivity
Shared data and services
"Back to the future"

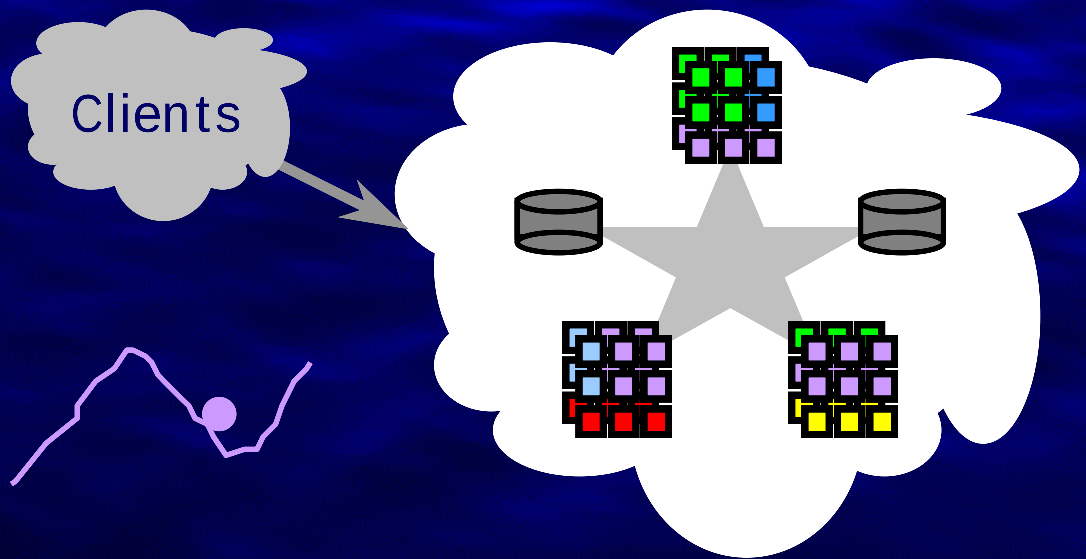
Compute clusters
Web farms, etc.



Network
storage
(e.g., SSP)

The Grid(s)
CDN or Utility
PlanetLab

Sharing the Server Cloud



E.g., CNN on 9/11
Facing a World Crisis
[William LeFebvre]

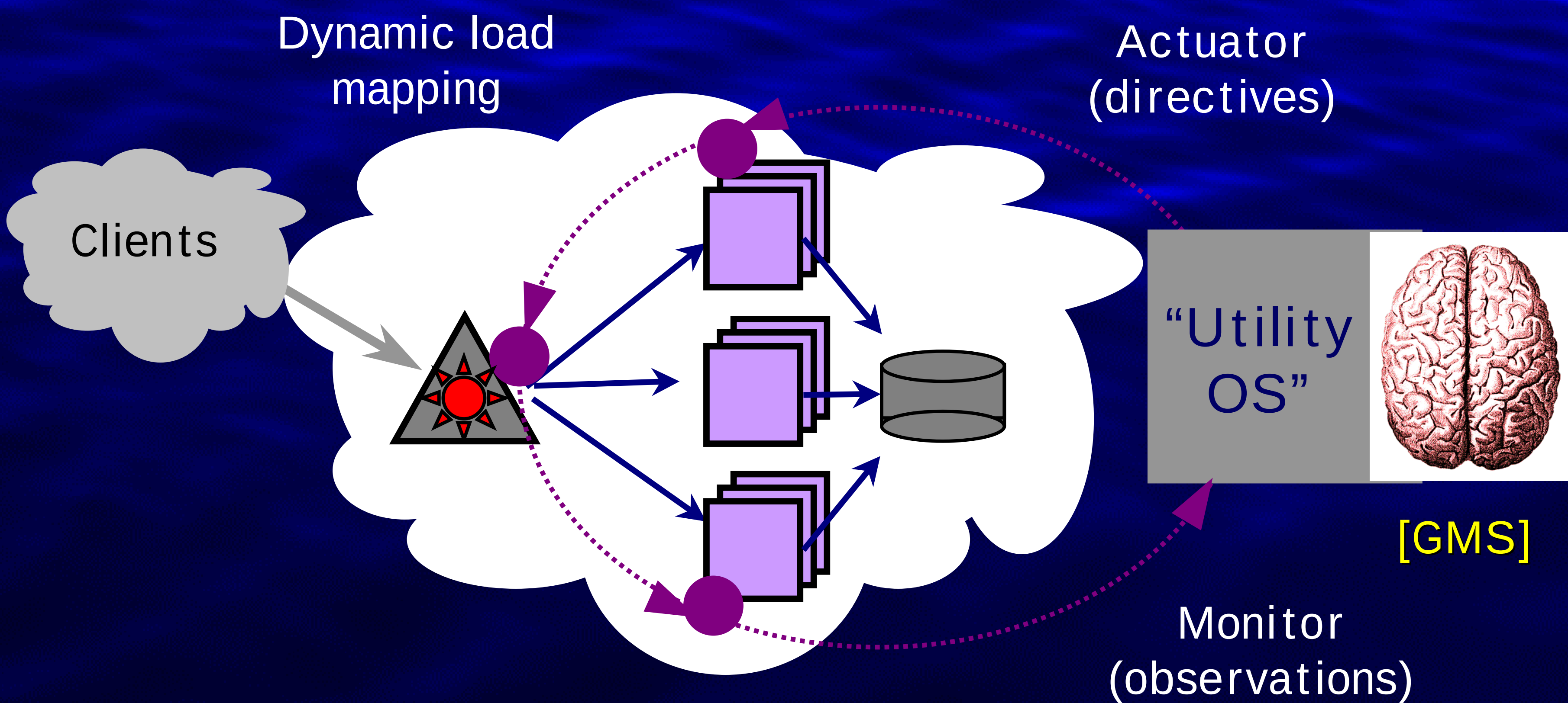
The world is dynamic, dangerous, unpredictable, and expensive.

- changing load/traffic
- changing demands
- unexpected failures
- unexpected load surges

Fluid mapping

Load multiplexing
Resource efficiency
Surge protection
Robustness
Incremental growth
Economy of scale

An OS for a Shared Server Cloud



Adaptive resource management

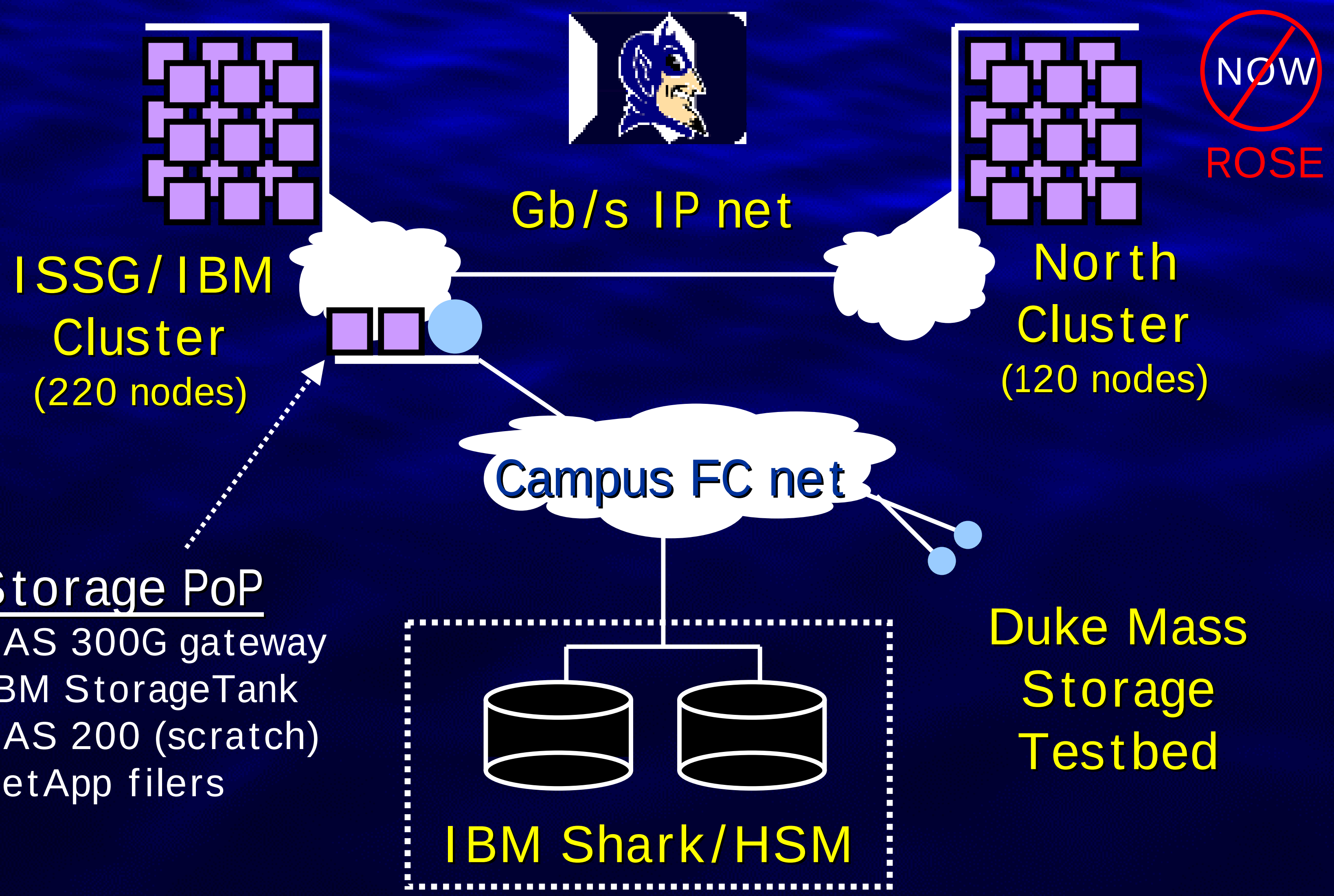
- Provisioning: how much?
- Placement: where?
- Isolation

Manage resources for target service quality levels (SLAs).

Outline

- Cluster-on-Demand (COD)
  - Decentralized resource management for mixed-use clusters
  - Automated configuration of dynamic virtual clusters
- Server provisioning for Web server utilities
  - [SOSP-01, USITS-03]
  - Dynamic thermal management (w/ HP Labs)
- Opus (Overlay Peer Utility Service) and ModelNet
  - Self-organizing network of server PoPs [OpenArch-02]
  - Evaluating adaptive Internet systems by emulation [OSDI-02]

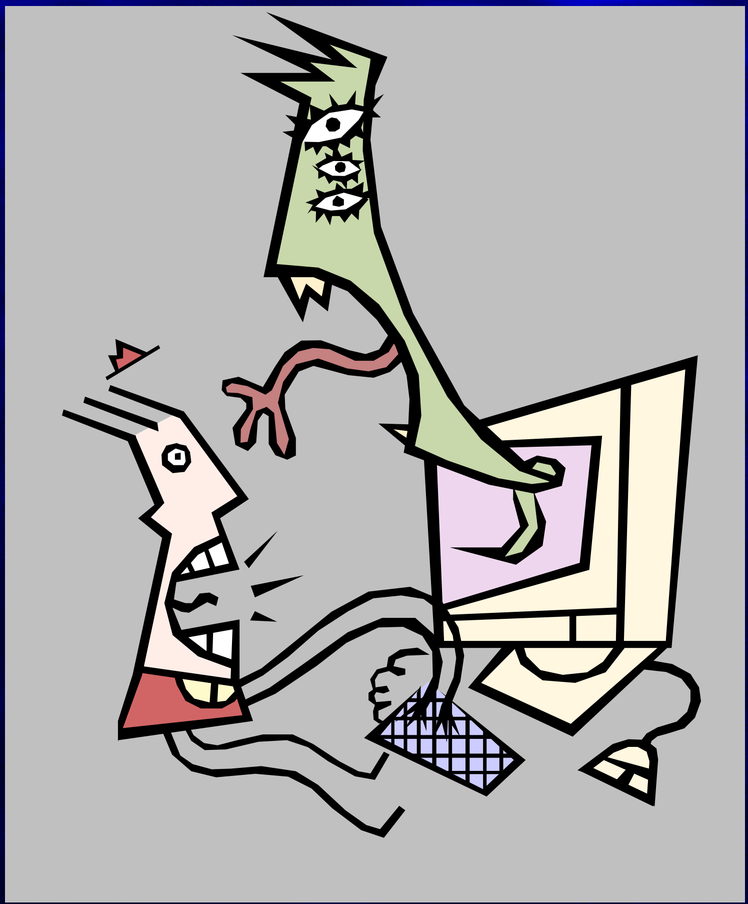
The Devil Cluster



Storage PoP

NAS 300G gateway
IBM StorageTank
NAS 200 (scratch)
NetApp filers

Life in Our Shared Cluster



Internet Systems and Storage Group

- Internet software prototypes
- Synthetic load generators
- ModelNet emulations
- Hacked kernels with bugs

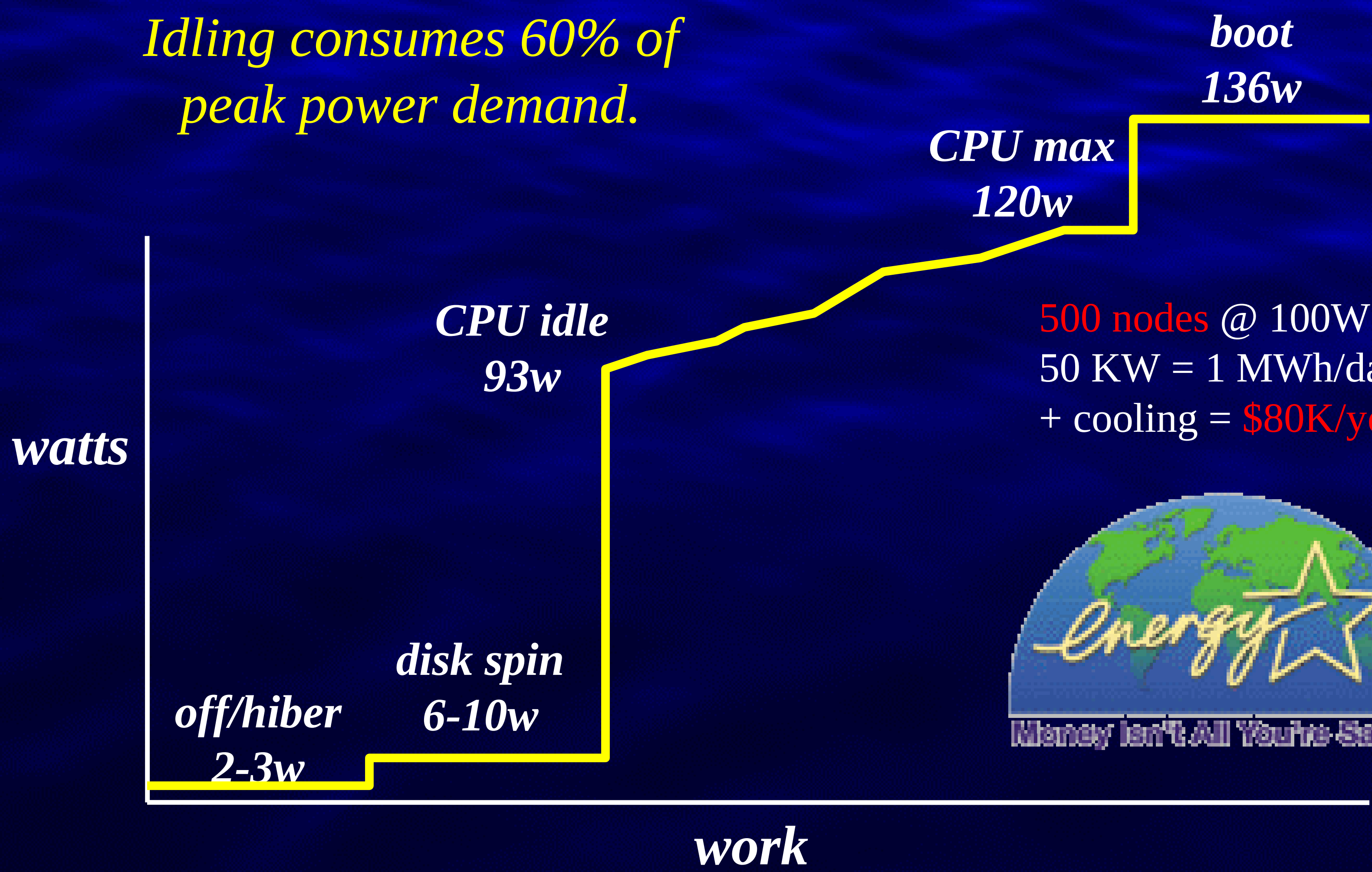
Computation Loads

- BioGeometry
- Full-system simulation
- Neuro-engineering simulations
- NC Biogrid PoP (more coming)
- Visualization

Heterogeneous servers
Diverse configurations
Manual reservation protocol
Bursty and deadline-driven

Server Energy

Idling consumes 60% of peak power demand.



500 nodes @ 100W
50 KW = 1 MWh/day
+ cooling = $80K/year

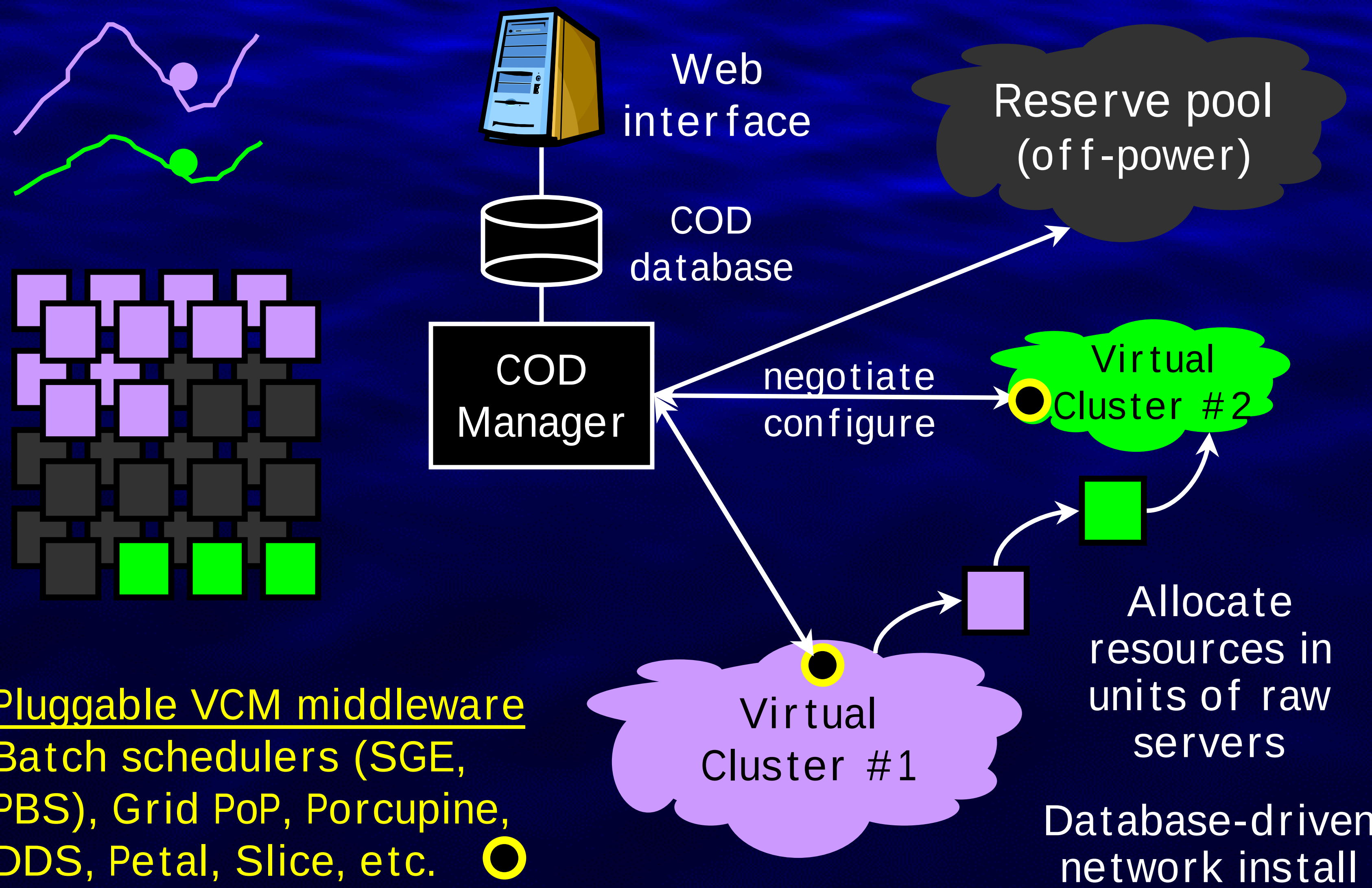
Money Isn't All You're Saving

866 MHz P-III SuperMicro 370-DER (FreeBSD), Brand Electronics 21-1850 power meter

Beyond Beowulf

- Compute clusters used to be hard to manage, but we've seen lots of progress since NOW.
  - Industry manageability initiatives (PXE, ACPI)
  - Configure/monitor/install: NPACI Rocks, Millennium/Rootstock.
  - Batch scheduling: LSF, PBS/Maui, SGE, etc.
  - Resource sharing/scavenging: Condor, Grids.
- Most assume a homogeneous software base and create a uniform cluster view for batch computation.
  - Beowulf model: Linux/Unix with middleware
- How to extend automated configuration and policy-based scheduling to mixed-use cluster utilities?
  - OS-agnostic? Middleware-agnostic?

COD: Dynamic Virtual Clusters



Pluggable VCM middleware

Batch schedulers (SGE, PBS), Grid PoP, Porcupine, DDS, Petal, Slice, etc.

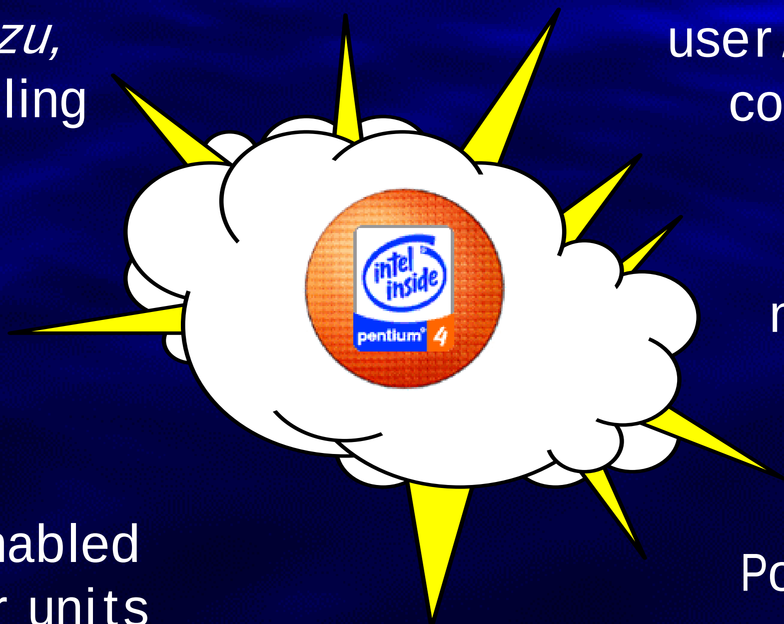
Enabling Technologies

Linux driver modules, Red Hat *Kudzu,* partition handling

DHCP host configuration

DNS, NIS, etc. user/group/mount configuration

PXE network boot



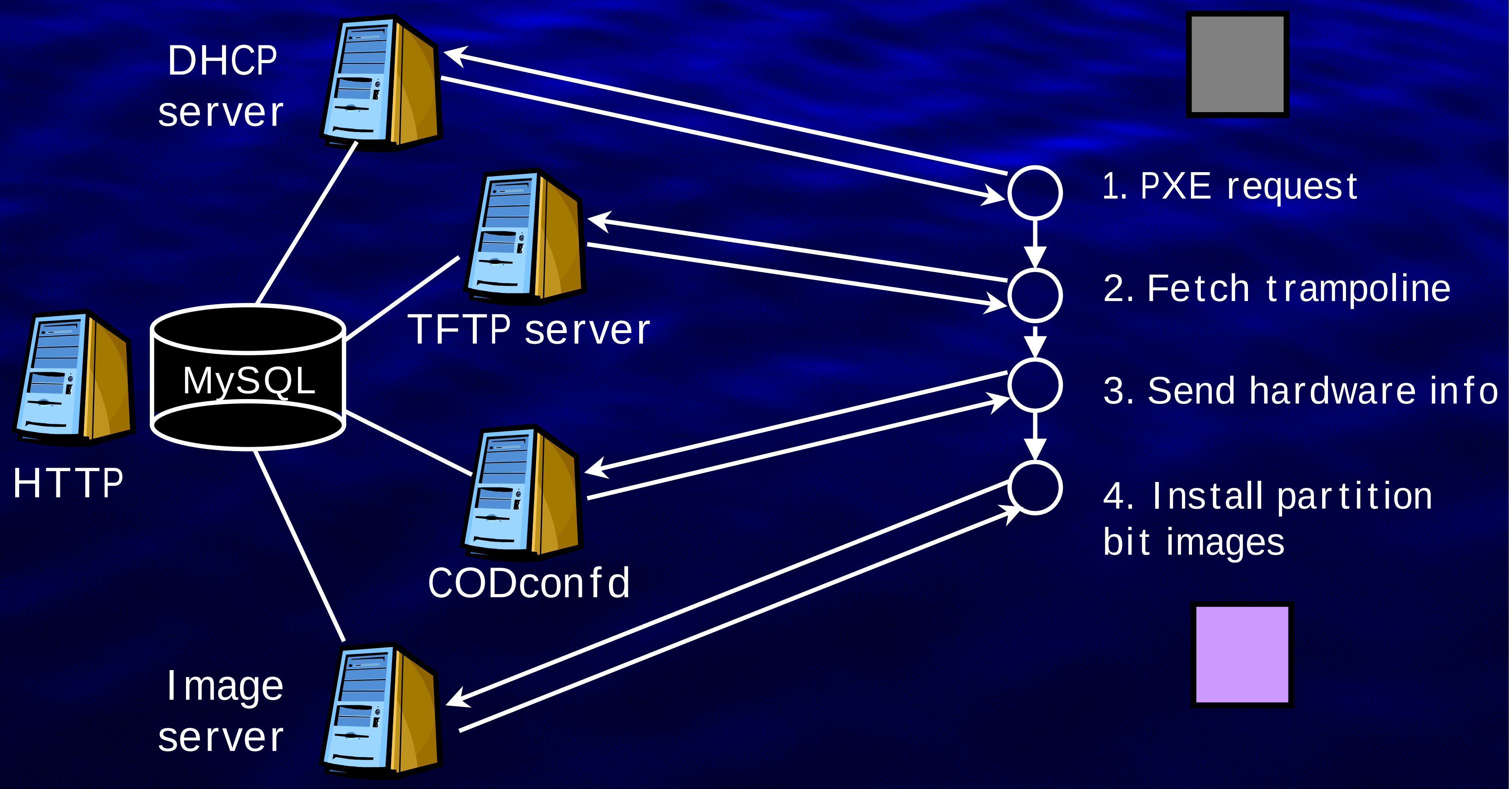
NFS etc. network storage automounter

IP-enabled power units

Ethernet VLANs

Power: APM, ACPI, Wake-on-LAN

COD Priming Steps



COD Priming Times

- Time to install images on one node
  - Image installation will dominate configuration time
- Node originally in “waiting” state
- Bottleneck: disk write speed

Image	Size (KB)	Size (gz)	% Full	Time (s)
CS Linux	2,048,256	105,456	23.2	122
CS BSD	2,048,256	137,020	27.4	129
Debian Linux	1,951,866	422,268	40.6	127

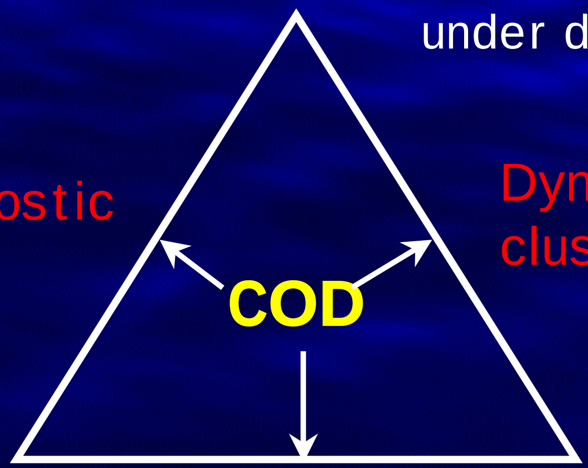
COD and Related Systems

Other cluster managers based on database-driven PXE installs:

Oceano hosts Web services under dynamic load.

OS-Agnostic

Dynamic clusters



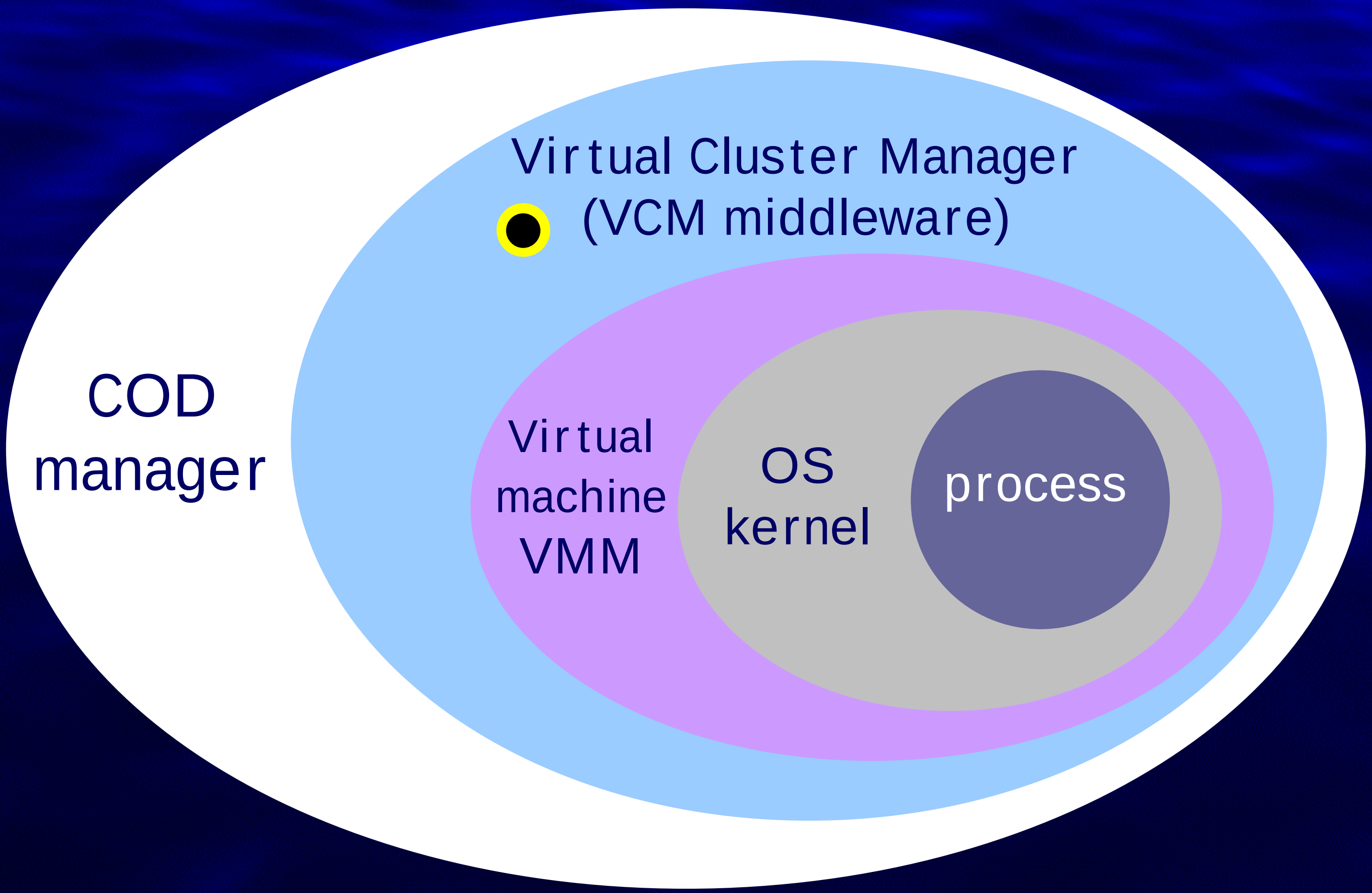
Netbed/Emulab configures static clusters for emulation experiments.

Flexible configuration
Open source

NPACI Rocks configures Linux compute clusters.

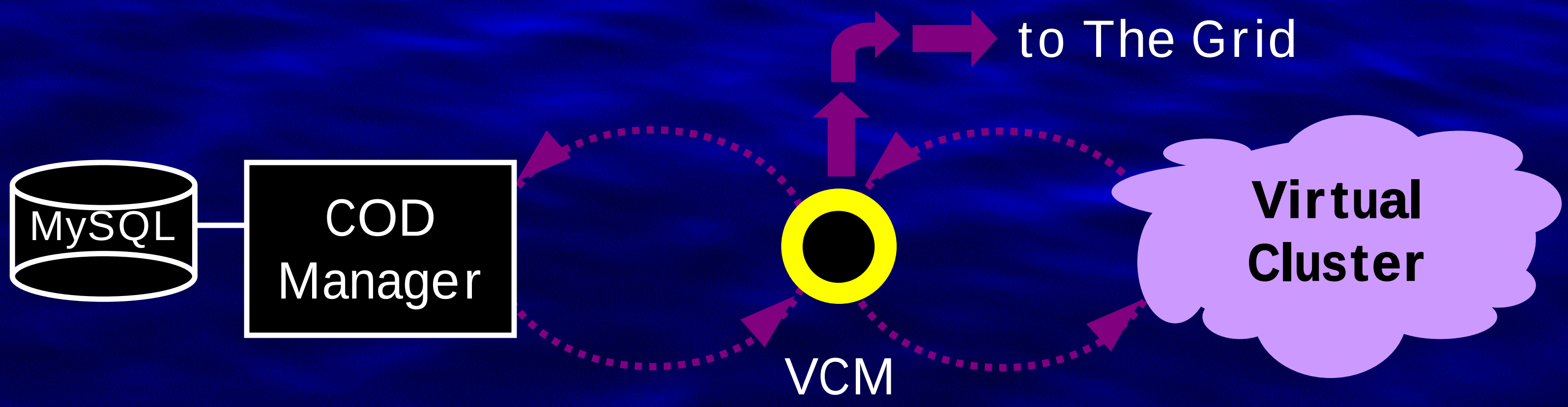
COD addresses hierarchical dynamic resource management in mixed-use clusters with pluggable middleware (“multigrid”).

A Note on Layering and Hierarchy



Like a VMM, COD manages resources outside/beneath the OS and without its knowledge or assistance: “underware”.

VCM Resource Negotiation

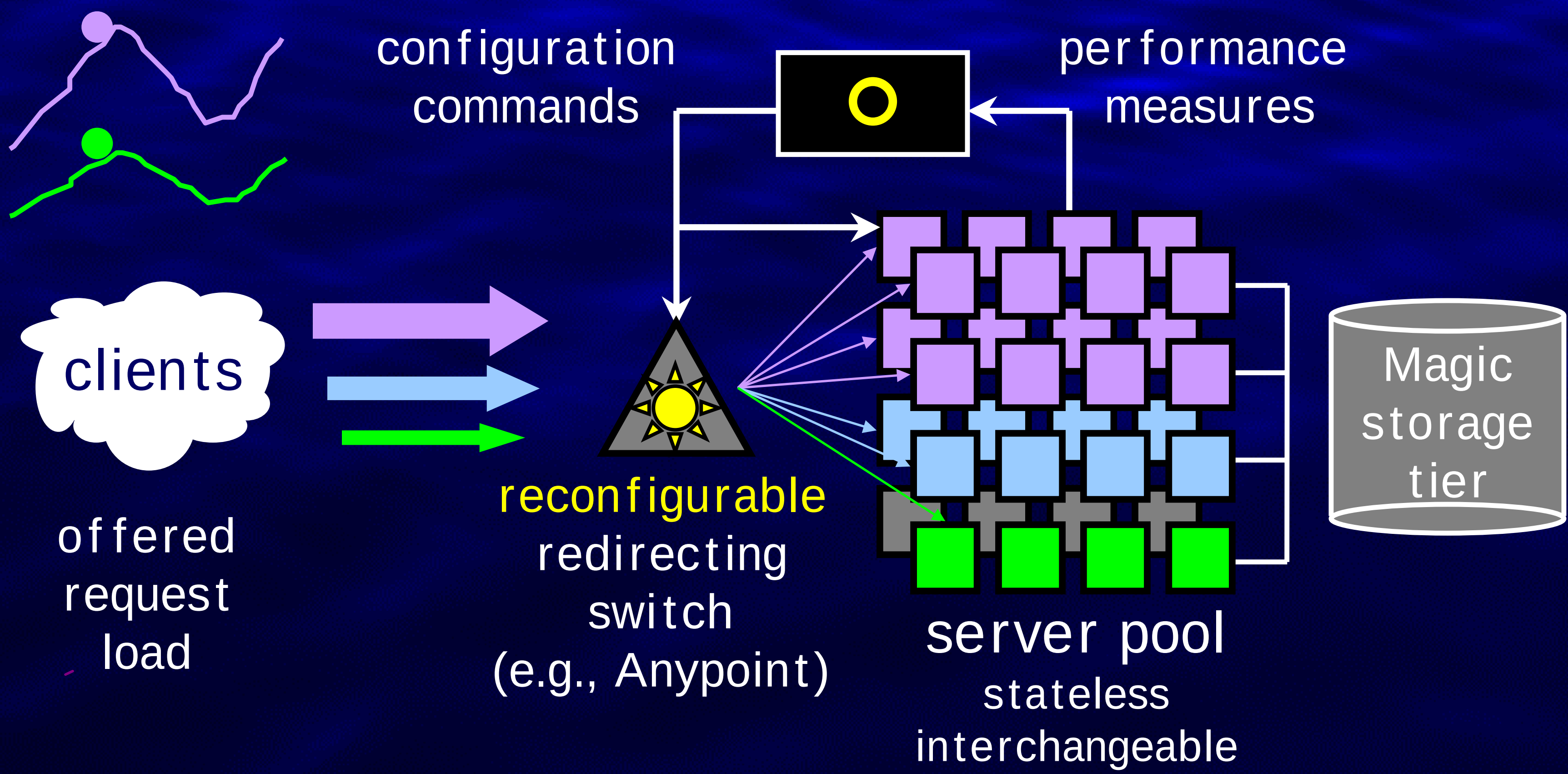


- Constraint-aware node assignment/negotiation
- Leased allotments for static clusters
- Load-aware provisioning for dynamic clusters
  - Balance local autonomy with global coordination
- Utility functions reflect value and priority (SLAs)
  - “Common currency” weight for VCM demand bids

Outline

- Cluster-on-Demand (COD)
  - Decentralized resource management for mixed-use clusters
  - Automated configuration of dynamic virtual clusters
- Server provisioning for Web server utilities
  - [SOSP-01, USITS-03]
  - Dynamic thermal management (w/ HP Labs)
- Opus (Overlay Peer Utility Service) and ModelNet
  - Self-organizing network of server PoPs [OpenArch-02]
  - Evaluating adaptive Internet systems by emulation [OSDI-02]

Muse: Adaptive Provisioning for a Web Service Utility [SOSP-01]

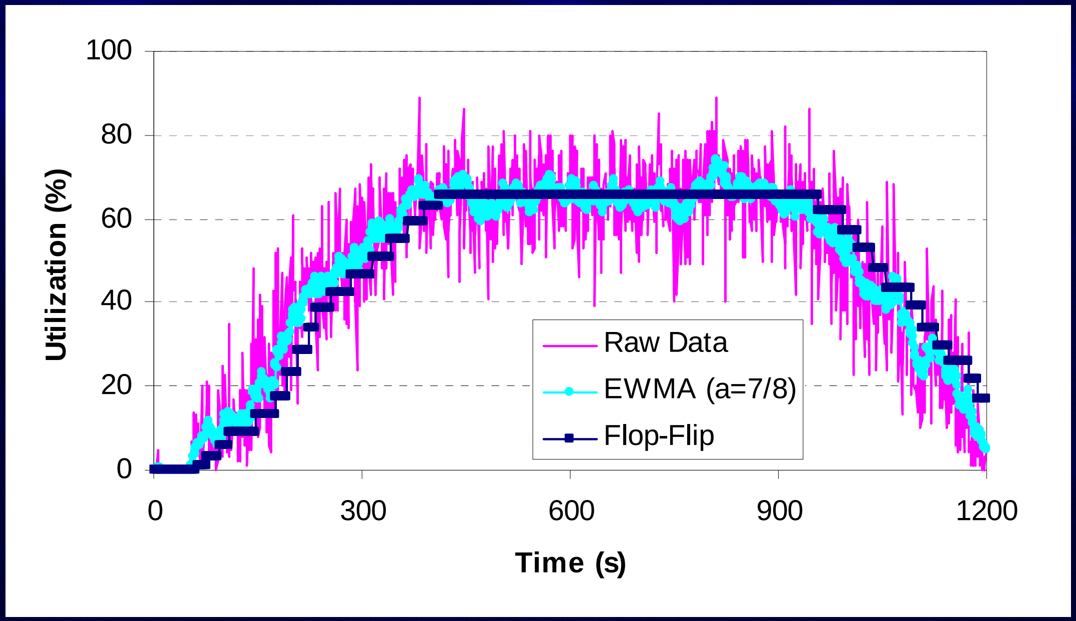


Multiplex services on nodes, based on proportional-share scheduling in node OS.

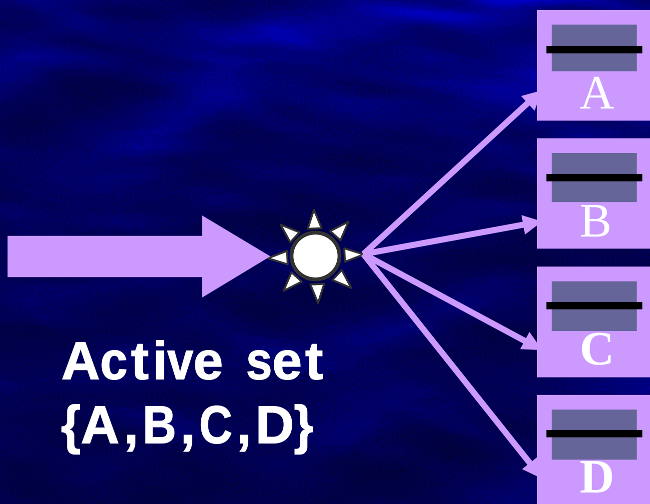
Filtering Load Estimates

- EWMA-based filter alone is not sufficient.
  - Average A_t for each interval t: $A_t = \alpha A_{t-1} + (1-\alpha)O_t$
  - The gain α may be variable or *flip-flop.*
- Load estimate $E_t = E_{t-1}$ **if** $|E_{t-1} - A_t| < tolerance$

  else $E_t = A_t$
- Stable
- Responsive



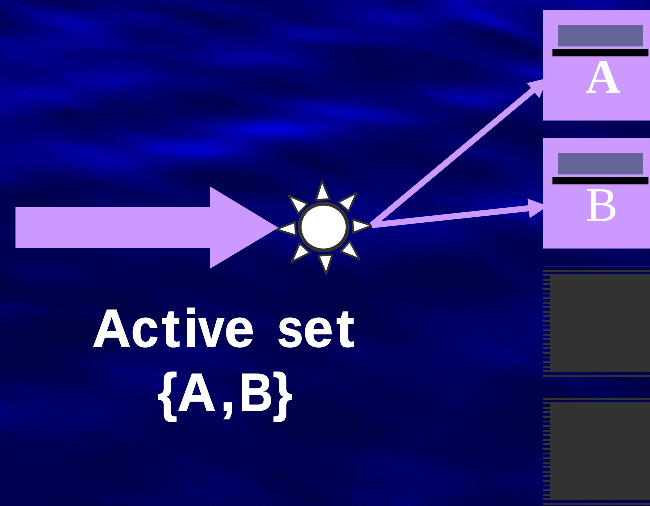
Cost vs. quality: a simple example



Active set {A,B,C,D}

$\rho_i < \rho_{target}$

- Low latency



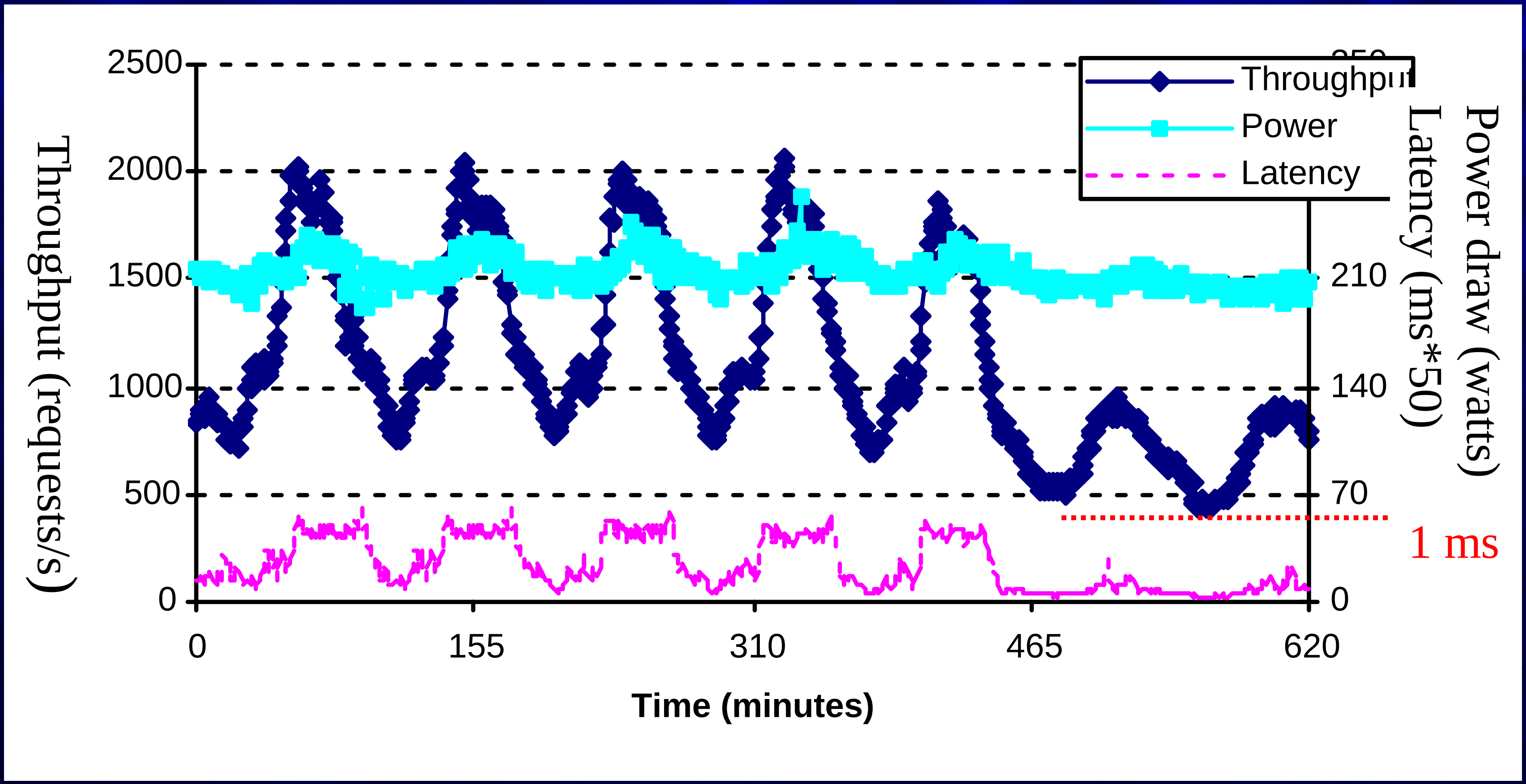
Active set {A,B}

$\rho_i = \rho_{target}$

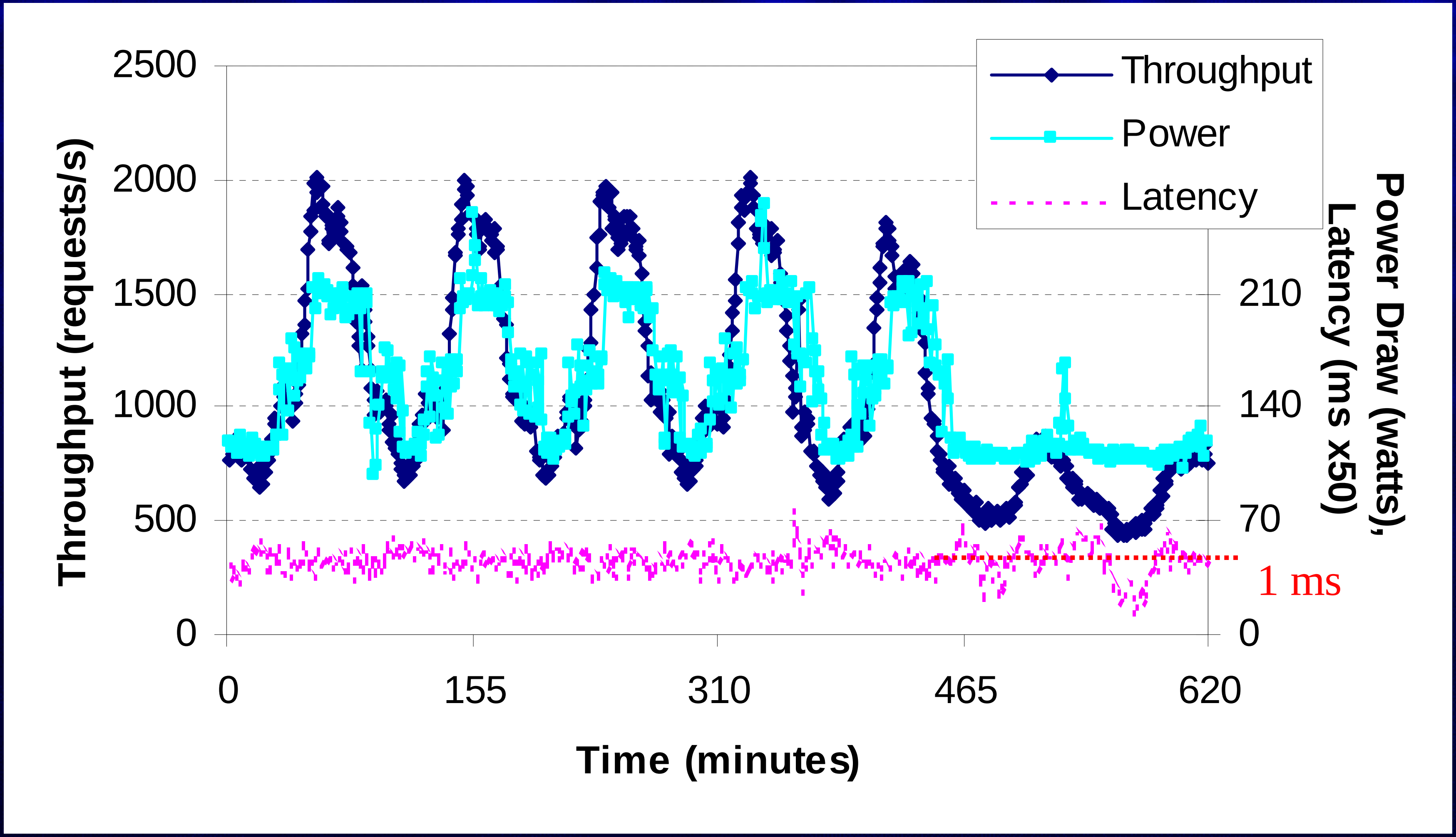
- Meets quality goals
- Efficient use

ρ_{target} = configurable target utilization (leaving headroom for transient load spikes).

IBM trace (*before*)



IBM trace (*after*)



Maximizing "Revenue" Under Constraint

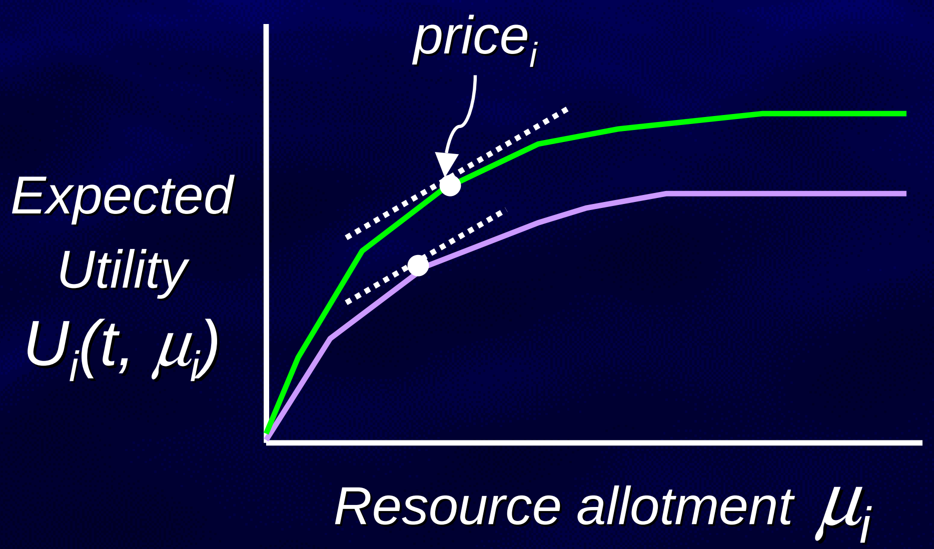
- Input: the "value" of *performance* for each customer *i*.
  - Common unit of value: "money".
  - Derives from the economic value of the service.
  - Enables SLAs to represent flexible quality vs. cost tradeoffs.
- Per-customer utility function U_i = *bid + penalty*.
  - Bid for traffic volume (throughput λ_i).
  - Bid for better quality, or add negative *penalty* for poor quality.
- Allocate resources to maximize expected global utility ("revenue" or reward).
  - Model predicts performance effects.
  - "Sell" μ to the highest bidder.
  - Never sell resources below cost.

Maximize $\sum U_i(\lambda_i(t, \mu_i))$

Subject to $\sum \mu_i < \mu_{max}$

SLAs as Utility Functions

- Customer SLAs are specified as utility functions
  - How much will customer "pay" for a given level of performance (or performability, data quality)?
  - Allocate resources to the highest predicted marginal benefit at current load and system conditions.



Dynamic Thermal Management

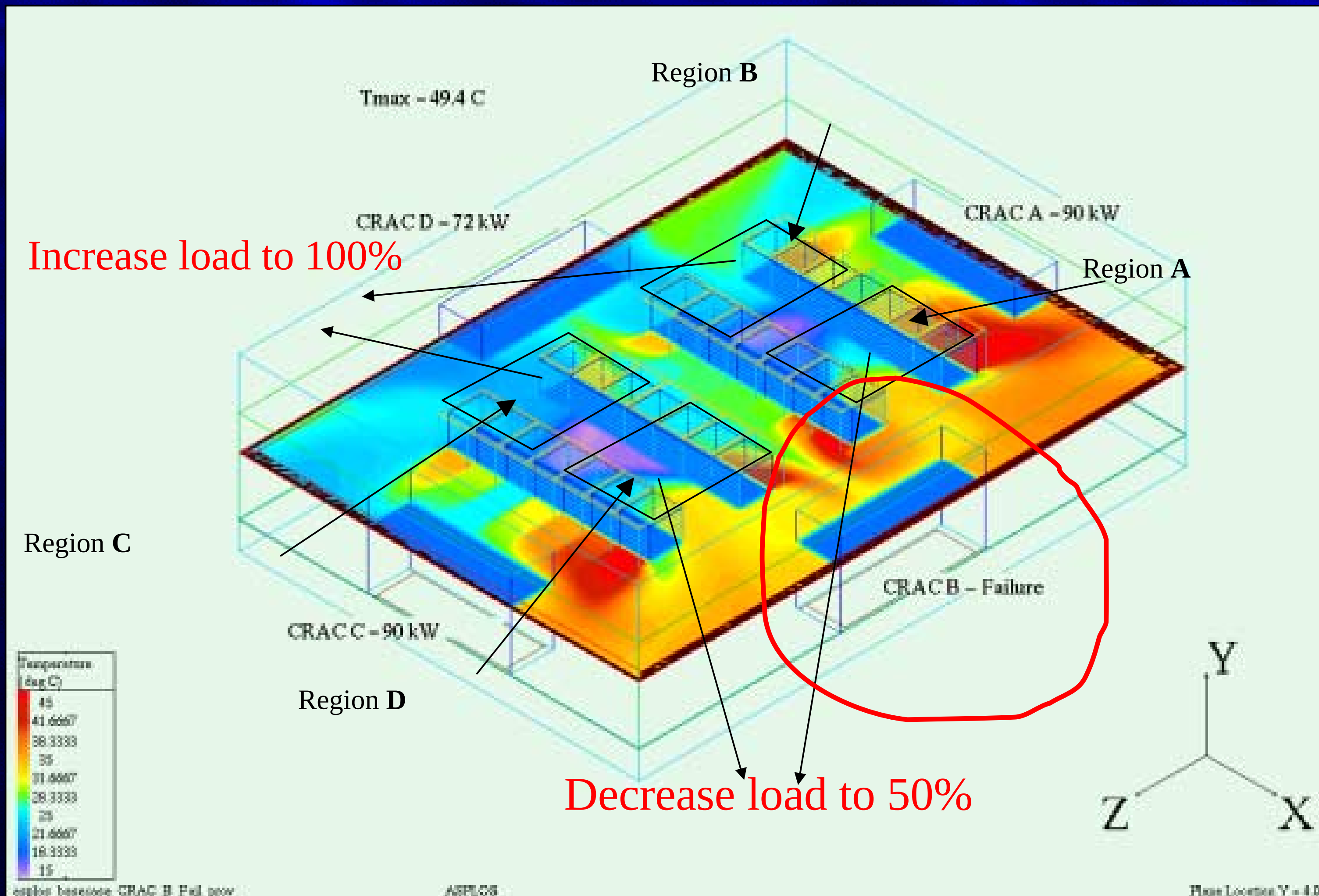
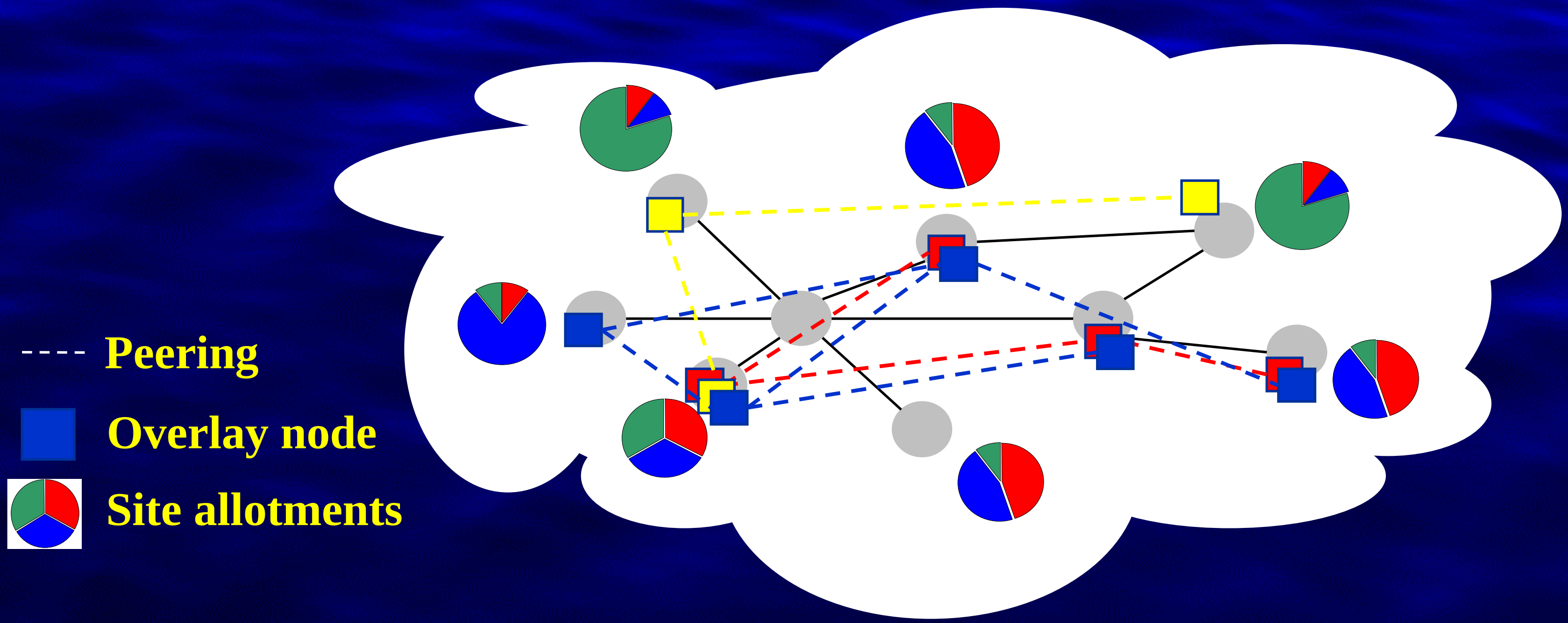


Figure 5. Temperature contour plot at 1.2m above floor: uniform workload, failed CRAC

Outline

- Cluster-on-Demand (COD)
  - Decentralized resource management for mixed-use clusters
  - Automated configuration of dynamic virtual clusters
- Server provisioning for Web server utilities
  - [SOSP-01, USITS-03]
  - Dynamic thermal management (w/ HP Labs)
- Overlay Peer Utility Service and ModelNet
  - Self-organizing network of server PoPs [OpenArch-02]
  - Evaluating adaptive Internet systems by emulation [OSDI-02]

Opus: Overlay Utility Service



Coordinated control of a network of server sites (PoPs)

Monitor and adapt for target performance and availability.

Configure/instantiate services on demand at selected sites.

Connect sites (e.g, replicas) with per-service overlay topology

- Metrics: bandwidth, latency, loss rate, cost ($) sensitivity

Decentralized management + weak global coordination

Evaluating Internet Systems

- Adaptive Internet software
  - Scalable, self-organizing, fast, robust,...
- How to evaluate prototypes?
  - How will it behave under dynamic conditions?
  - How will it behave when the unexpected occurs?
- Simulation
  - There is no substitute for build-and-measure.
- Live deployment
  - Runs real code under real (uncontrolled) conditions: dangerous and expensive, does not yield reproducible results.

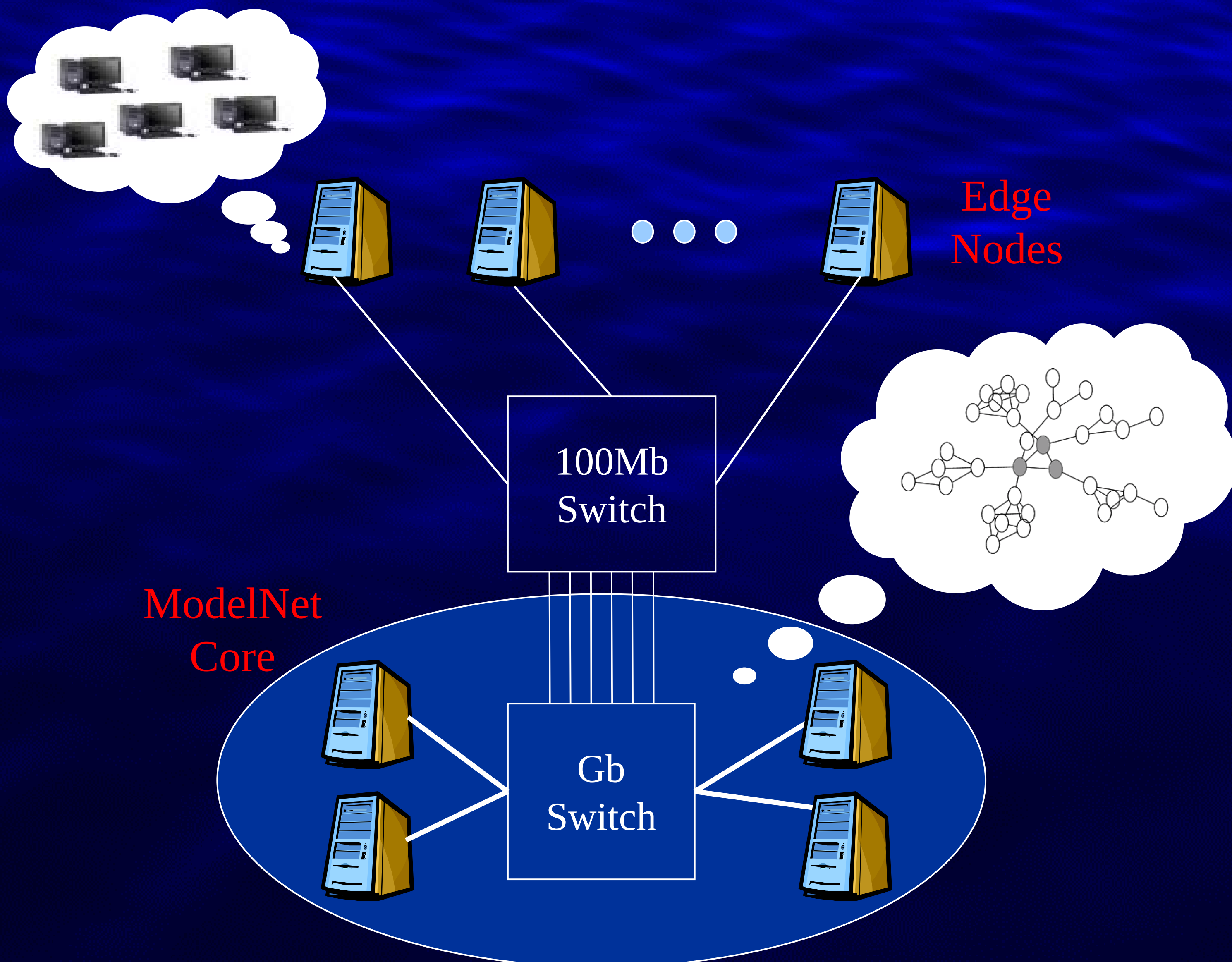
The ModelNet Vision

- Emulation environment for adaptive, large-scale services
  - Web services, multimedia distribution, p2p, mobile systems
  - Accelerate development of robust, adaptive Internet software
- Challenges
  - Scalable to 10k nodes, 10 Gb/s bisection bandwidth
    - On 100-node cluster
    - Unmodified applications run on unmodified OS's
  - Accurately emulate wide-area network conditions
    - Failures, cross traffic, rapid changes, congestion

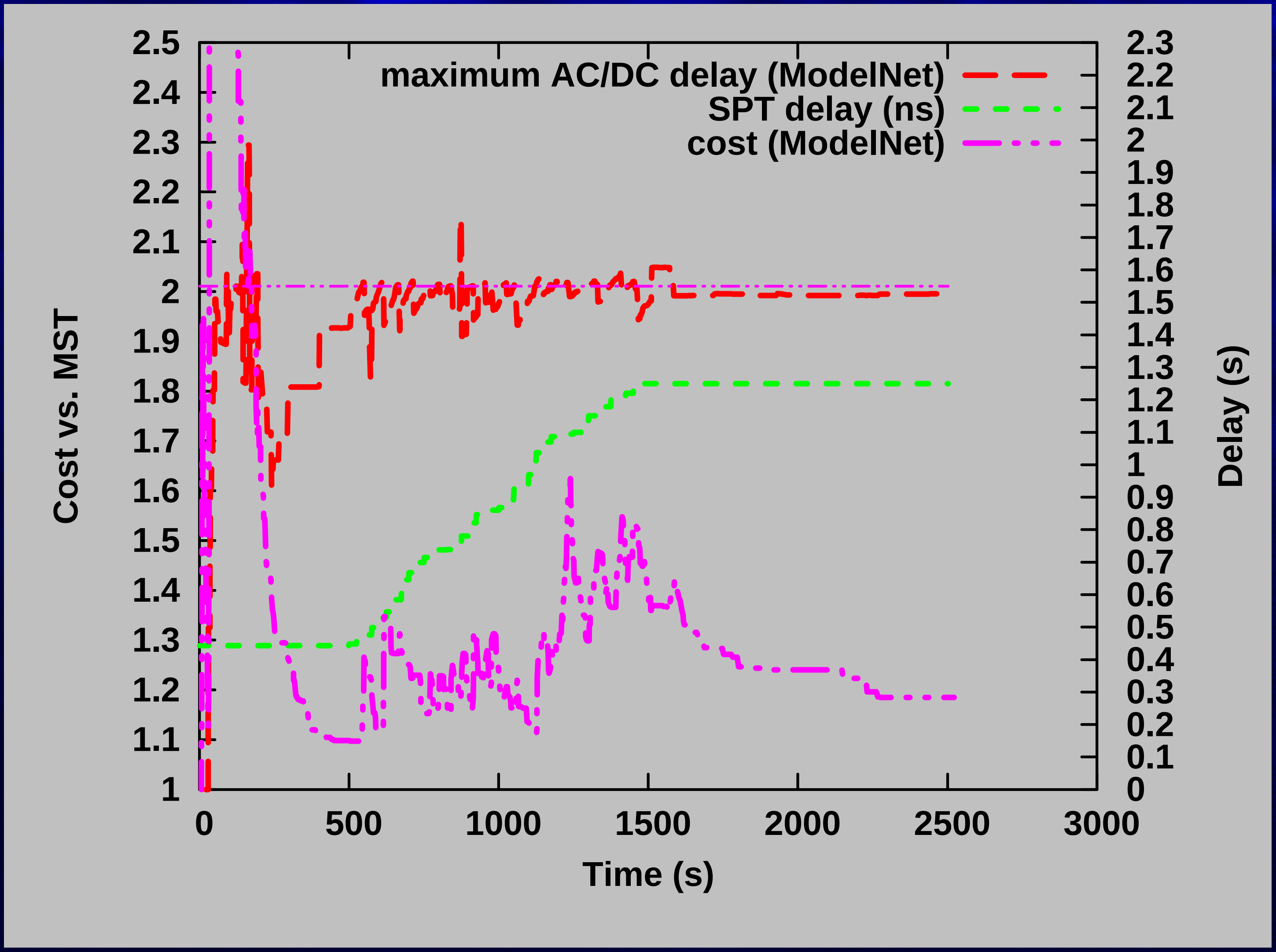
➔ Impossible to capture full complexity of Internet in a machine room

  - Application-specific accuracy vs. scalability tradeoffs

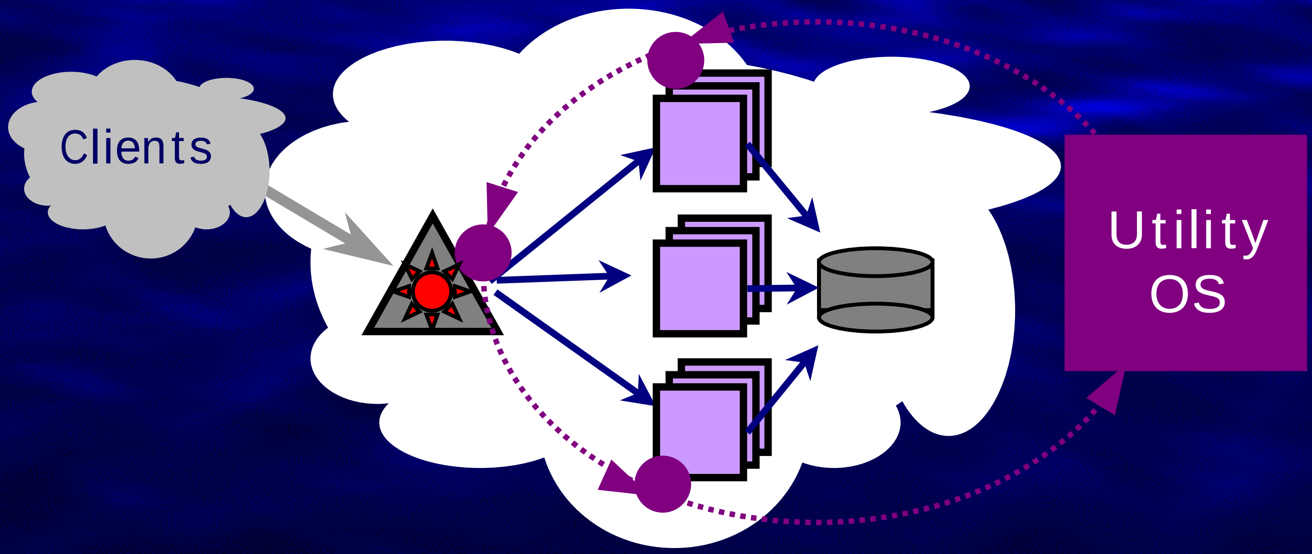
ModelNet



Self-Organizing Overlay



Server Utility in a Dynamic World
A Few Key Challenges



- Multiple goals and dimensions of service quality (SLAs)
- Coordinated provisioning of multiple resources that interact in a complex way
- Configure service spread and location for performability
- Reconcile local autonomy and global control

http://issg.cs.duke.edu

http://www.cs.duke.edu/~chase