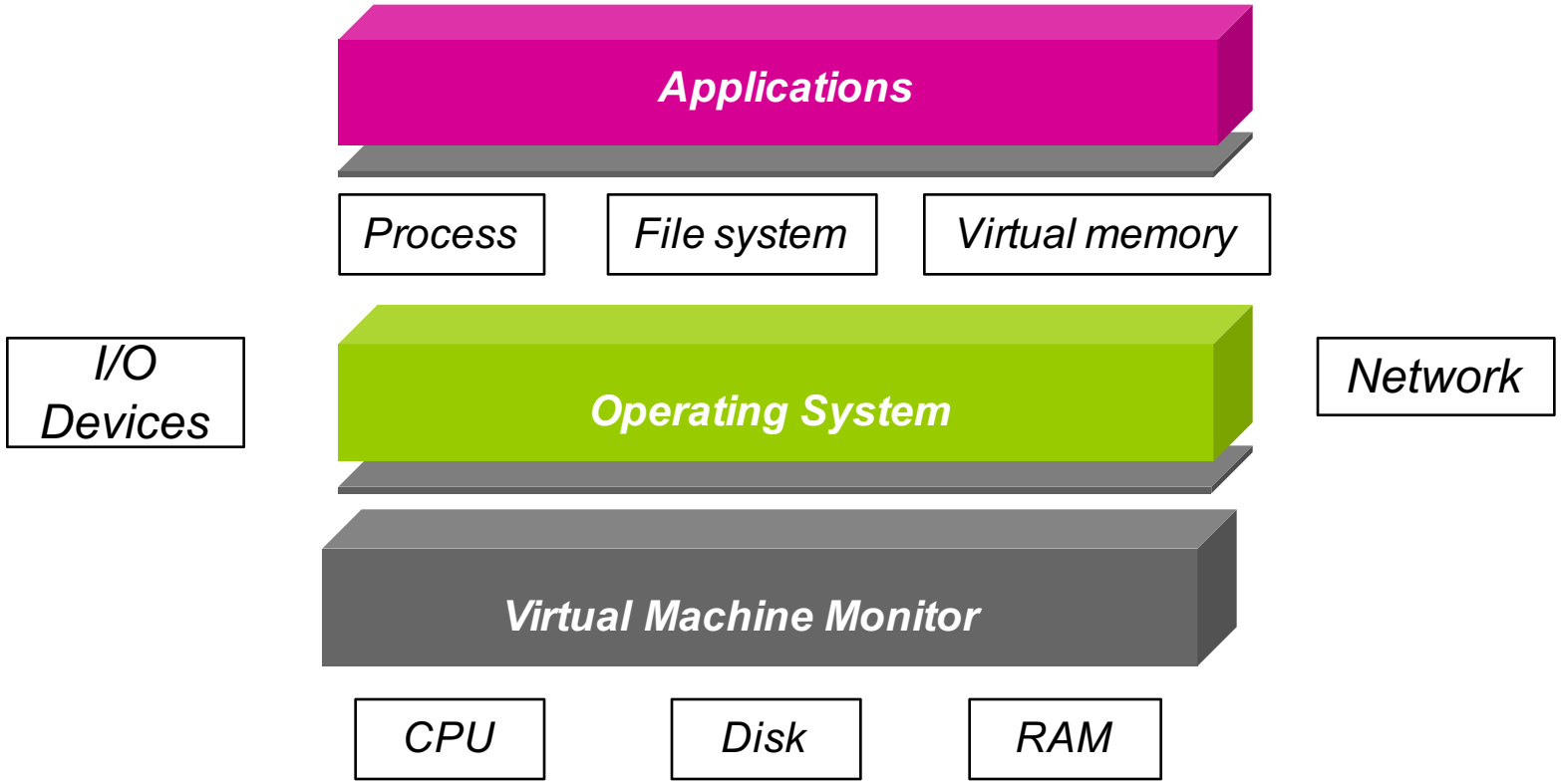
Virtual Machine Monitor



What is a VMM?

- OS offers illusion that each process is running on its own computer

- A VMM virtualizes an entire physical machine
  - VMM offers illusion that OS has full control over the hardware
  - VMM "applications" (OSes) run in virtual machines

- Implication: Can run multiple instances of different OSes simultaneously on a machine

Why do such a crazy thing?

- Resource utilization
  - Machines today are powerful, multiplex their hardware
  - Migrate VMs across machines without shutdown
- Software use and development
  - Can run multiple OSes simultaneously
    - No need to dual boot
  - Can do system (e.g., OS) development at user-level
- Many other cool applications
  - Debugging, emulation, security, fault tolerance, …

Cool VMM Tricks

- How to experiment with apps, protocols, and systems on future hardware?
  - Example: How to experiment with 100 Gbps network?
- Time dilation
  - VMM slows timer interrupt to make hardware (CPU, disk, network) appear faster to OS and apps
  - Example:
    - OS reads 10 Gb of data from network in 1 second, but thinks only 0.1 second has elapsed
    - But, applications run 10x slower

VMM Requirements

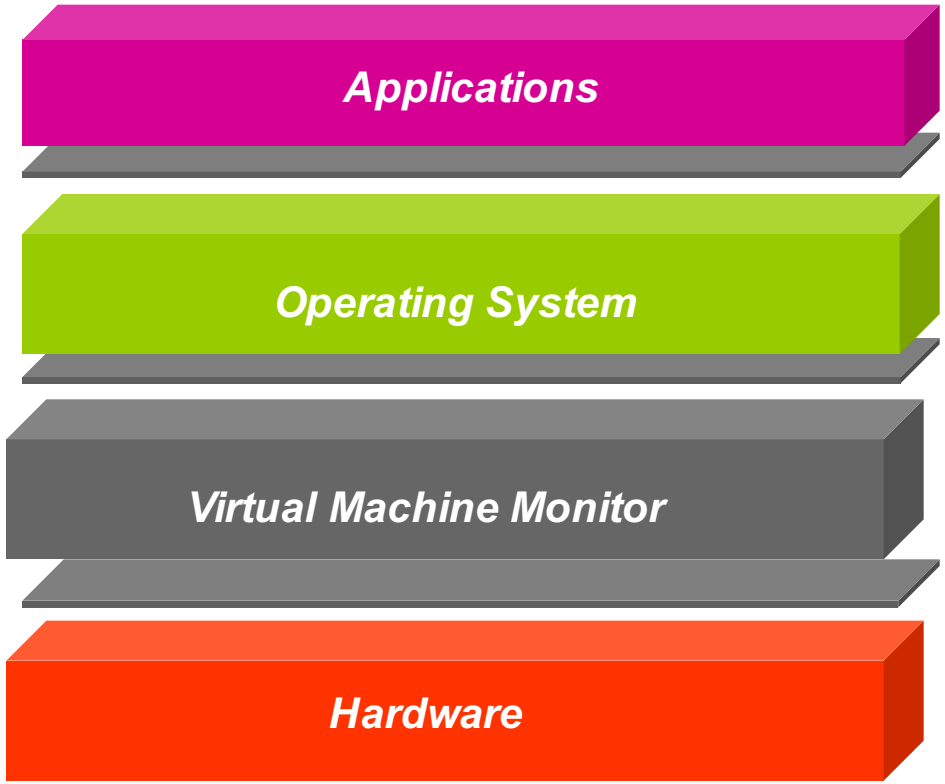
- Fidelity
  - OSes and applications work without modification
    - (although we may modify the OS a bit)
- Isolation
  - VMM protects resources and VMs from each other
- Performance
  - VMM is another layer of software → overhead
    - As with OS, want to minimize this overhead

VMware

- VMware workstation uses hosted model
  - VMM runs unprivileged, installed on base OS
  - Relies upon base OS for device functionality
- VMware ESX server uses hypervisor model
  - VMM runs directly on hardware

- VMware uses software virtualization
  - Dynamic binary rewriting translates code executed in VM
  - Rewrite only privileged instructions to reduce overhead

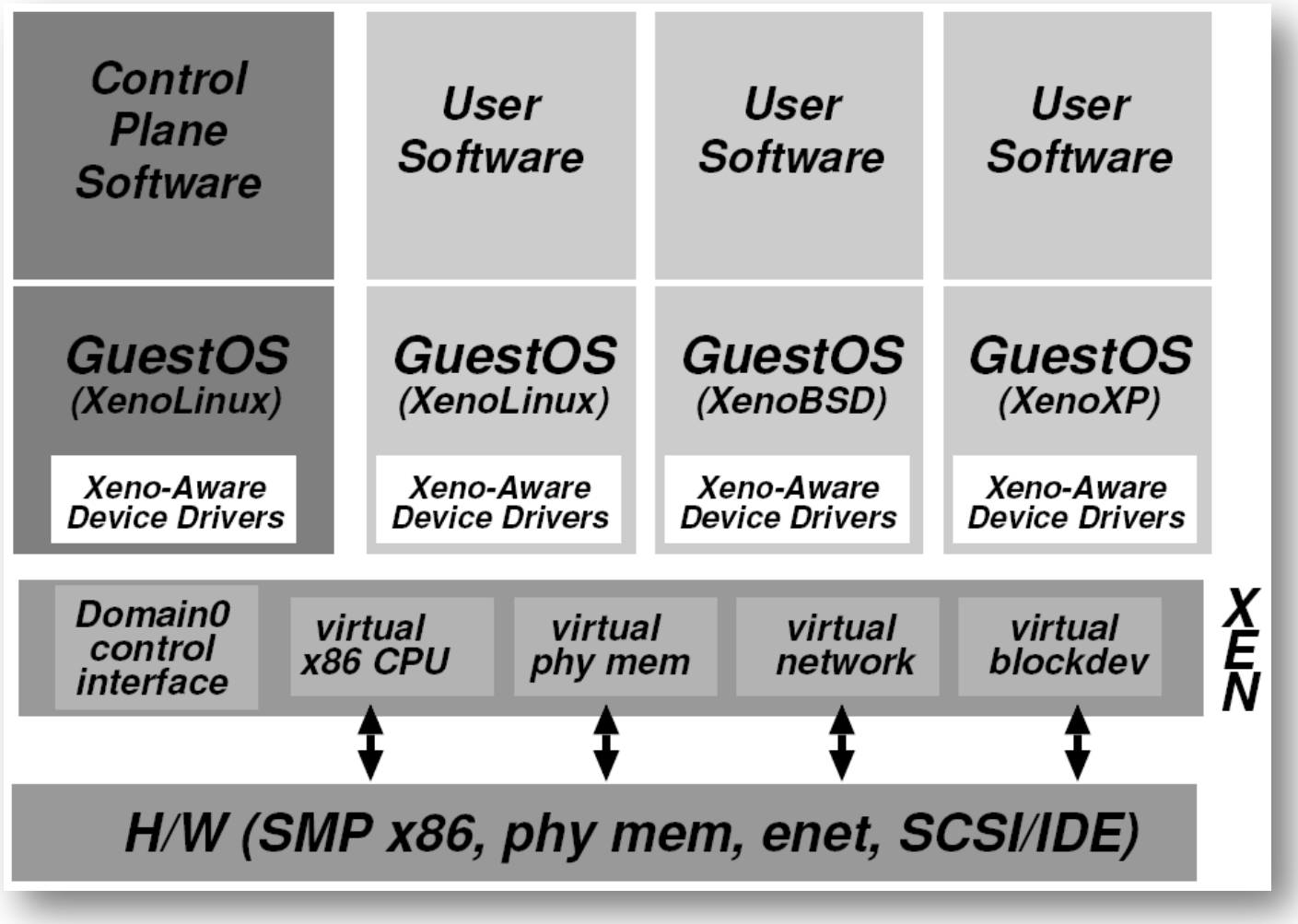
VMware Hypervisor Model



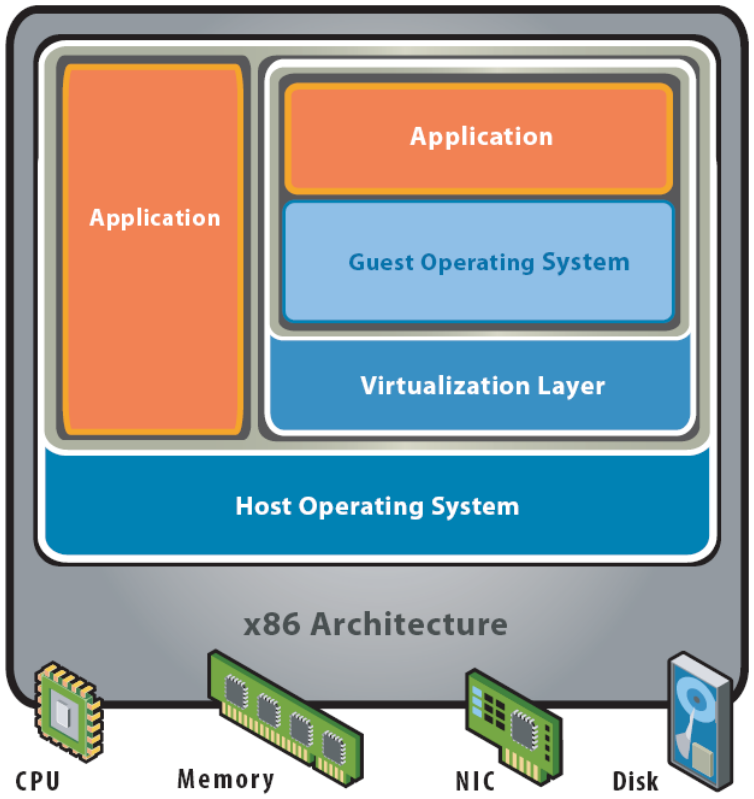
Xen

- Early versions use "paravirtualization"
  - Fancy word for "we have to modify & recompile OS"
- Xen hypervisor (VMM)
  - VMM runs at privilege, VMs (domains) run unprivileged
  - Trusted OS (Linux) runs in own domain (Domain0), manages privileged operations
- Most recent version does not require OS mods
  - Because of Intel/AMD hardware support
- Commercialized via XenSource, but also open source

Xen Architecture



VMware Hosted Architecture



Hosted Architecture

What needs to be virtualized?

- Exactly what you would expect
  - CPU
  - Events
  - Memory
  - I/O devices
- Isn't this just duplicating OS functionality?
  - Yes and no
  - Approaches will be similar to what OS does
    - Simpler functionality (VMM much smaller than OS)
  - But implements a different abstraction
    - Hardware interface vs. OS interface

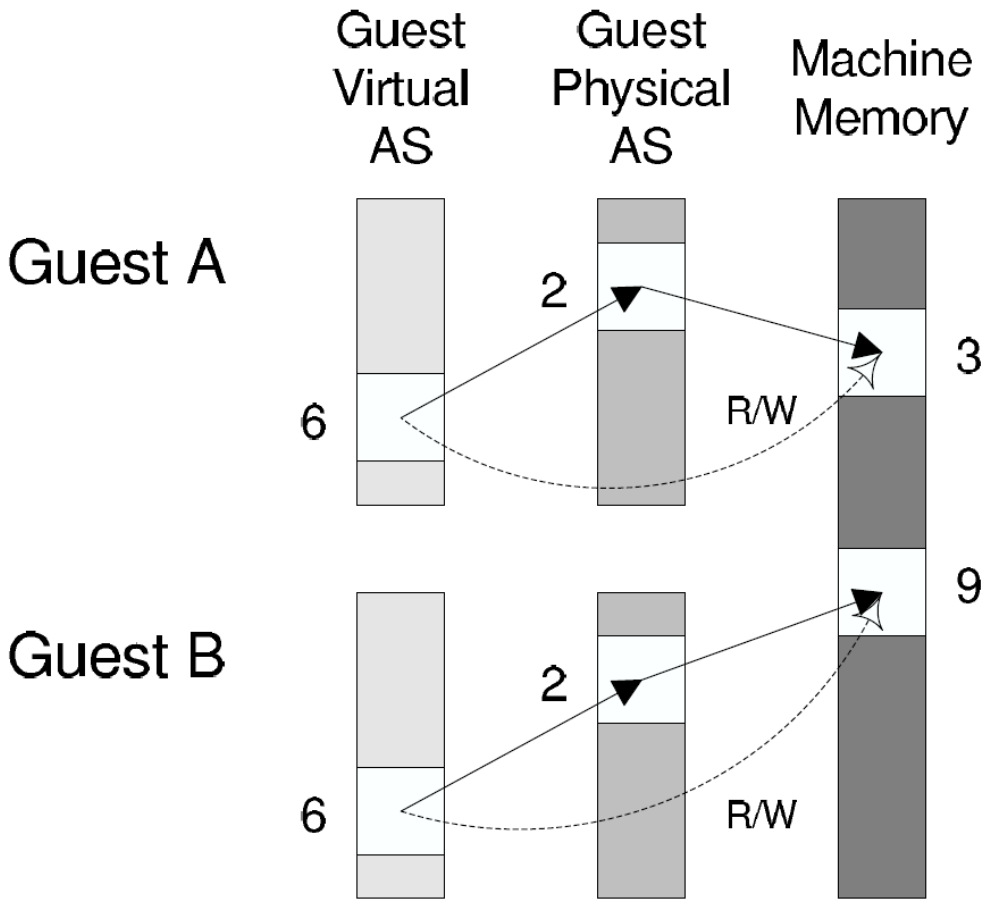
Virtualizing Memory

- OS assumes full control over memory
- But VMM partitions memory among VMs
  - VMM needs to control mappings for isolation
    - OS can only map to a physical page given to it by VMM
- Solution: Need MMU support to handle two-levels of page tables

Shadow Page Tables

- Three abstractions of memory
  - Machine: actual hardware memory
    - 2 GB of DRAM
  - Physical: abstraction of hardware memory managed by OS
    - If a VMM allocates 512 MB to a VM, the OS thinks the computer has 512 MB of contiguous physical memory
    - (Underlying machine memory may be discontiguous)
  - Virtual: virtual address space per process
    - Standard 2^{32} address space
- In each VM, OS creates and manages page tables for its virtual address spaces without modification

Shadow Page Tables



The Cloud and Datacenters

- *Where* is the web?

Services

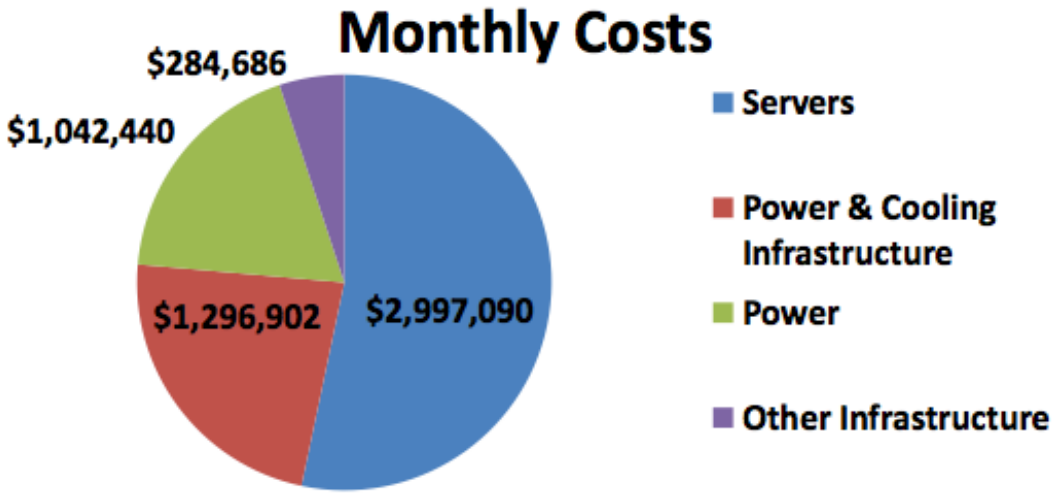
Security	Biometric scan, locked doors	Guard, cards, faraday cages
Environment	A/C!	
Network	Multiple Physical Taps	Multiple Networks
Power	Transformers	Diesel, batteries, flywheels
Weather hardening	Roof!	Earthquake resistance

Datacenter Economics

- Some datacenters run by a single entity
  - Google, Microsoft, University of Michigan
- Others rent space to smaller customers
- Cost of running datacenter combination of:
  - Amortized costs of the building (15 yr)
  - Amortized costs of electrical equipment
  - Amortized costs of servers (3 yr)
  - Power, Bandwidth, Labor
- When you rent space, you pay for:
  - Space, power, bandwidth, maybe labor

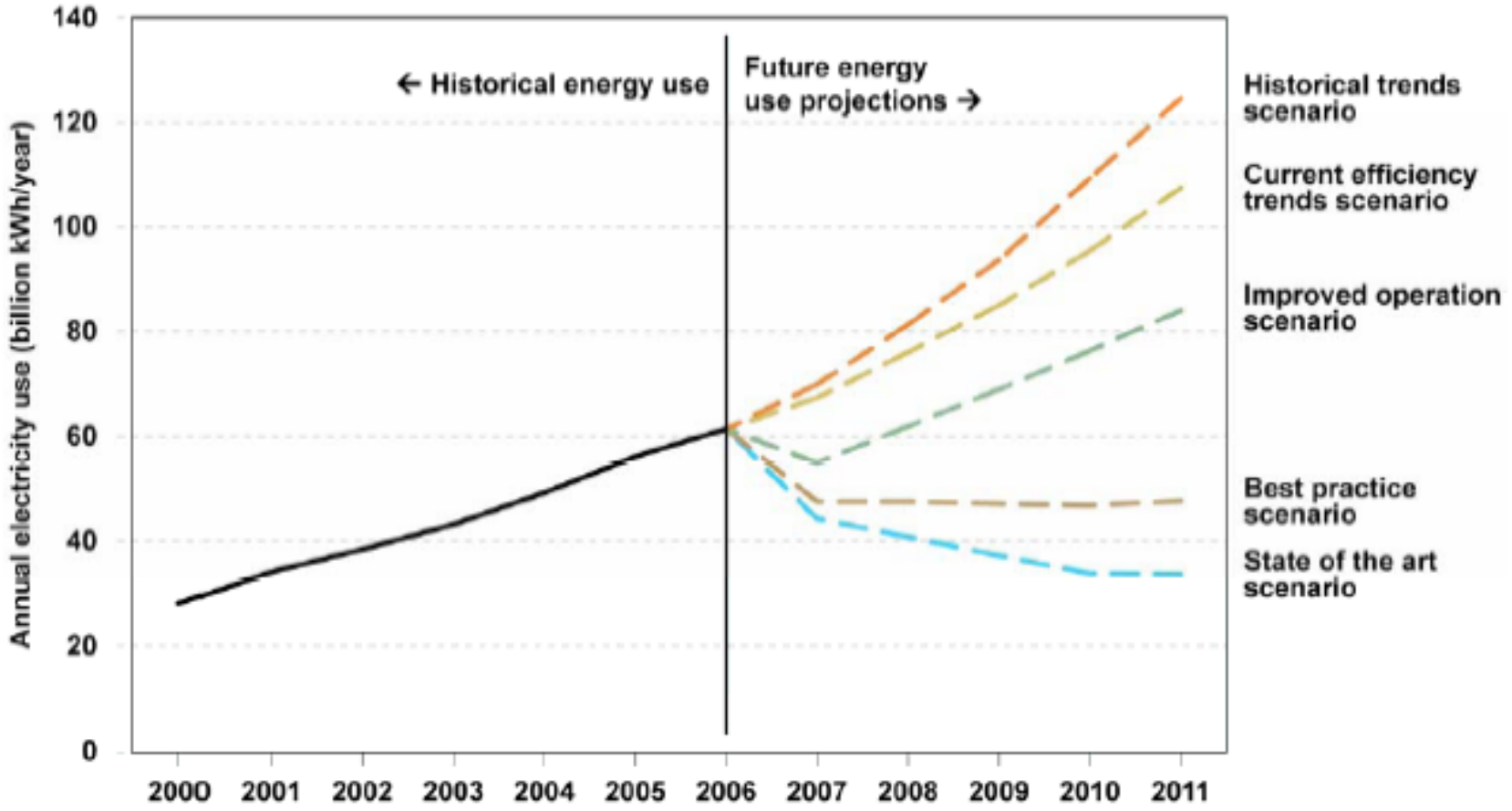
Power

- Some numbers from James Hamilton
  - Facility: ~$200M for 15MW facility
  - Servers: ~$2k/ea, about 50k of them
  - Power draw at 30% util: 80%
  - Commercial power: 0.07/kWhr



3yr server & 15 yr infrastructure amortization

Datacenter Energy Growth

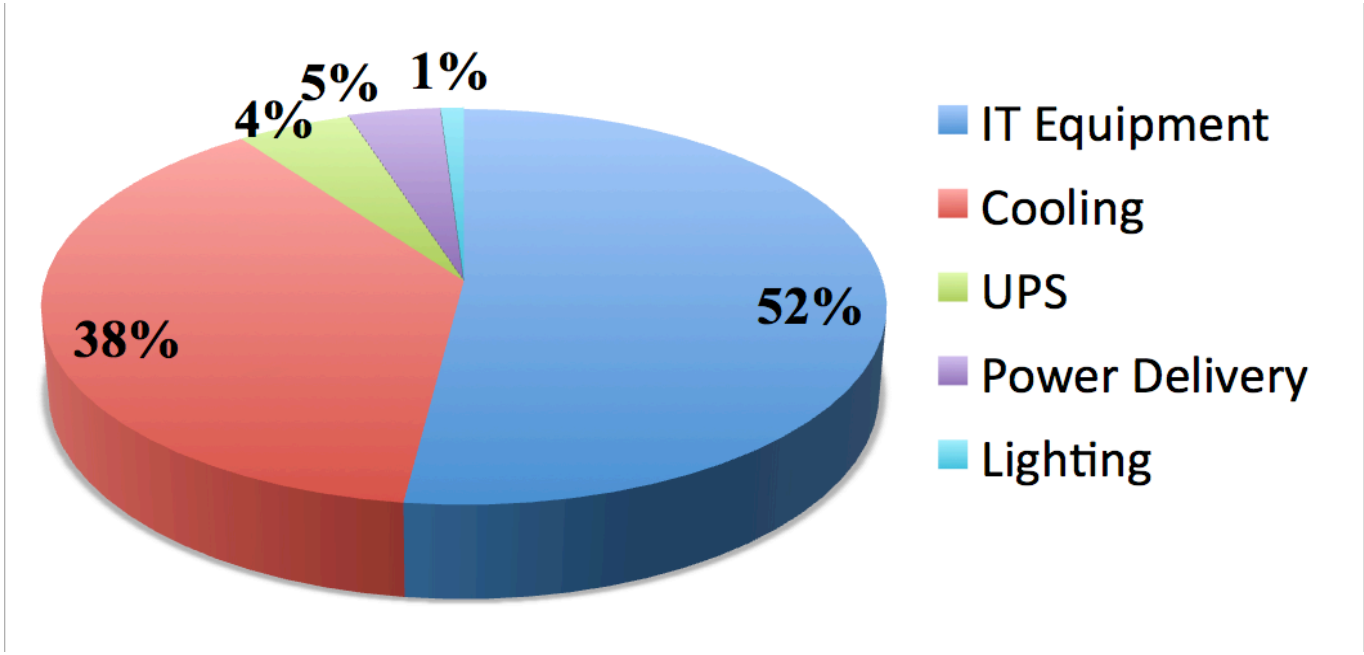


- Very old data, but… installed base grows 11%/year
- In 2012, 2% of all US energy use

Fully-burdened cost of power

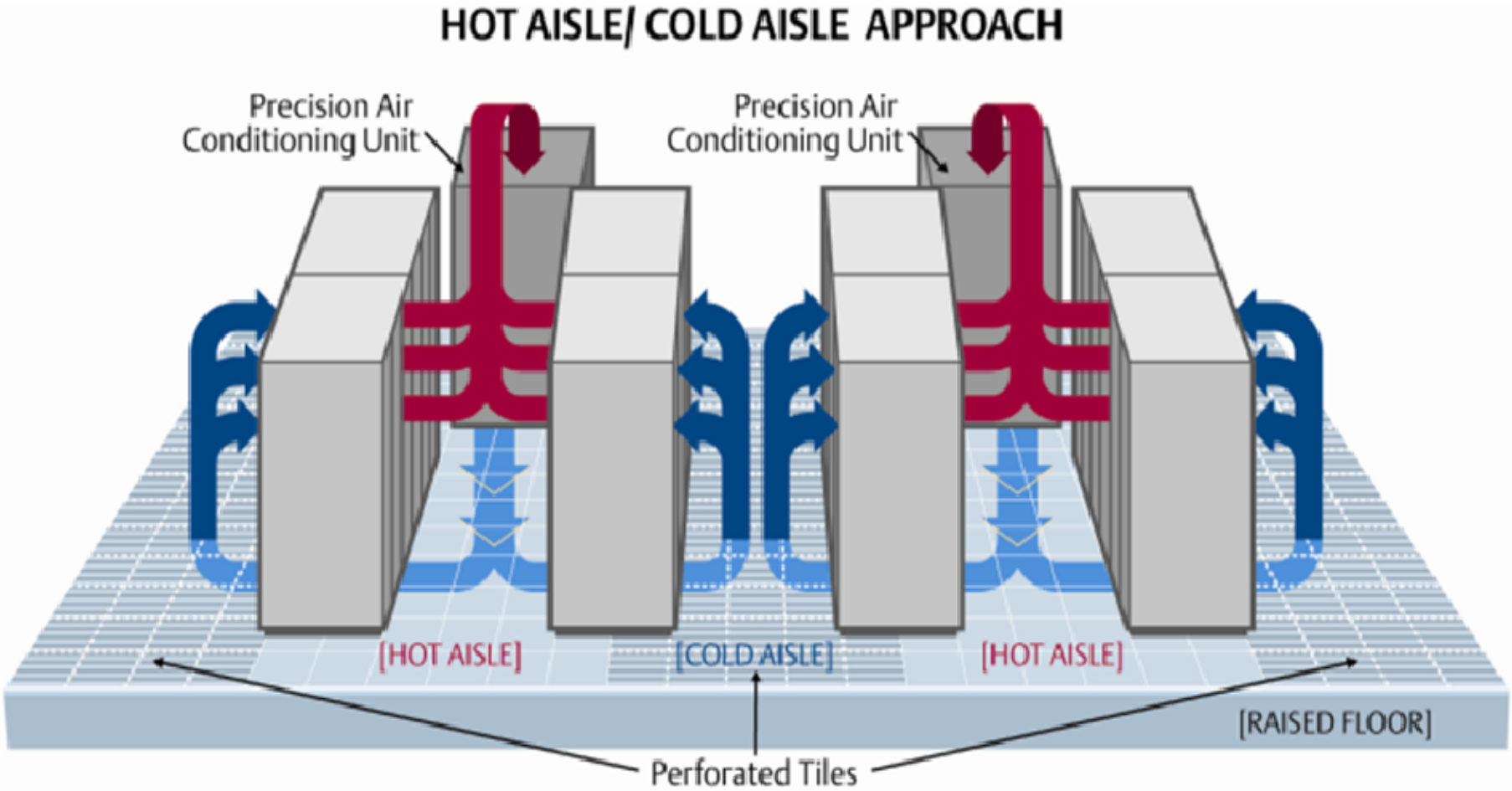
- Infrastructure cost/watt
  - Using 15-year amort, 5% interest
  - $1.28/W/yr
- Energy cost of watt using 0.07 Kwhr
  - $0.83/W/yr
- Full cost: $2.11/W/hr

Heat



- Servers account for barely half of power
  - 1W of cooling per 1.5W of IT load
- Managing energy consumption means, to a large extent, managing heat

Cooling the datacenter



Other Sources of Inefficiency

- Energy wasted by idle systems
  - Up to 75% idleness; but idle power ~60% of peak
- Inefficient cool air distribution
  - Hot exhaust recirculation halves cooling efficiency
- Power conversion losses
  - 4+ conversions; some under 70% efficiency
- Poor server performance per watt
  - Embedded sys. up to 5*x* better on Web 2.0 apps [Lim '08]

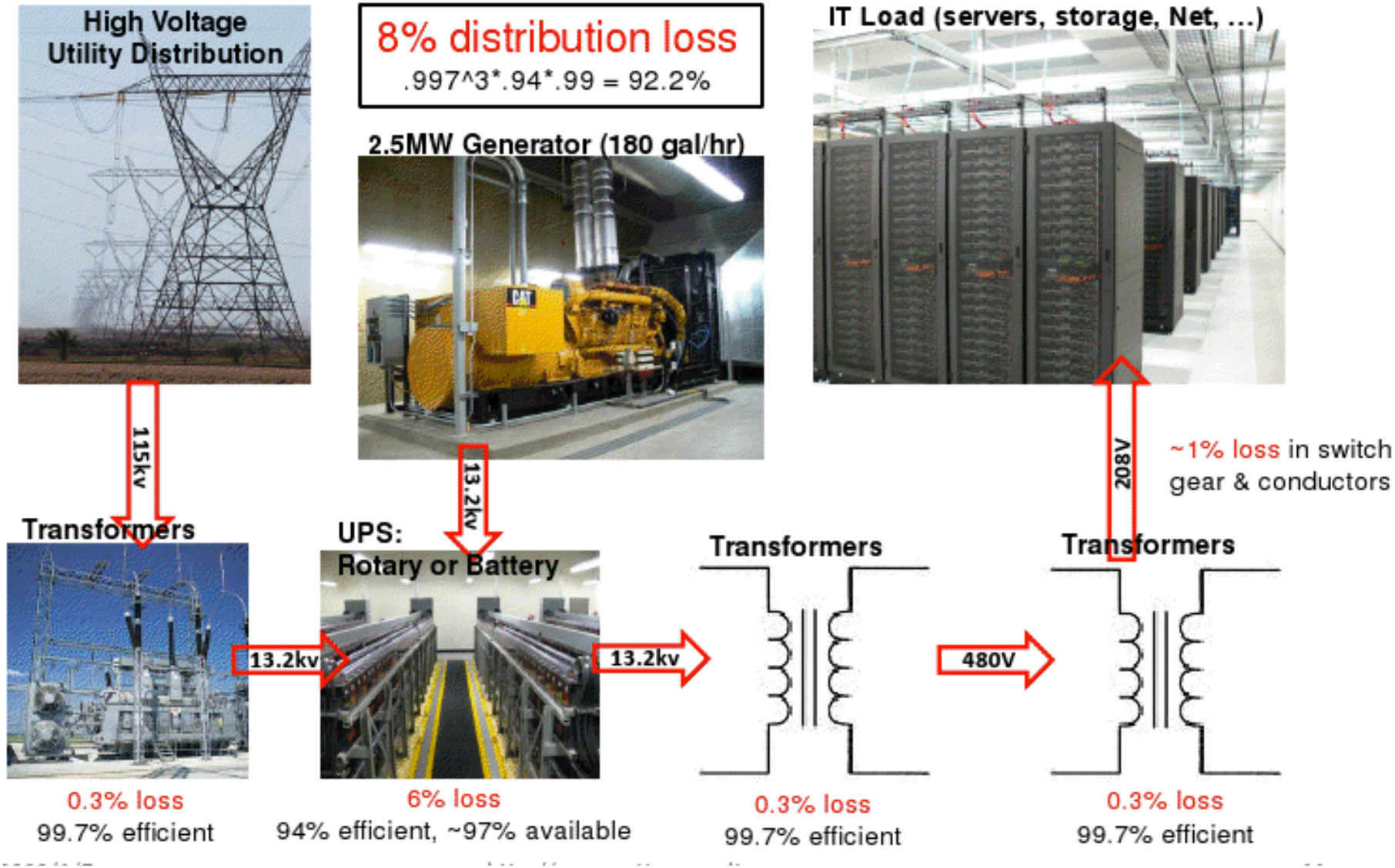
Datacenter Efficiency

- Power Usage Effectiveness (PUE)
  - Total Facility Power / IT Equipment Power
  - For each watt, how much to computing?
  - 1.0 means no extra cost at all
  - Facebook claims 1.07
  - The Michigan datacenter we tour is ~2, but others on campus are 1.1
- In principle, you can get <1 by recycling waste heat

Current Challenges

- Whole datacenter must be optimized, not just filled with more efficient computers
  - CPUs only account for 12% of energy!
- Challenges:
  - Power infrastructure very inefficient
  - Utilization & poor energy-proportionality
  - Cooling efficiency

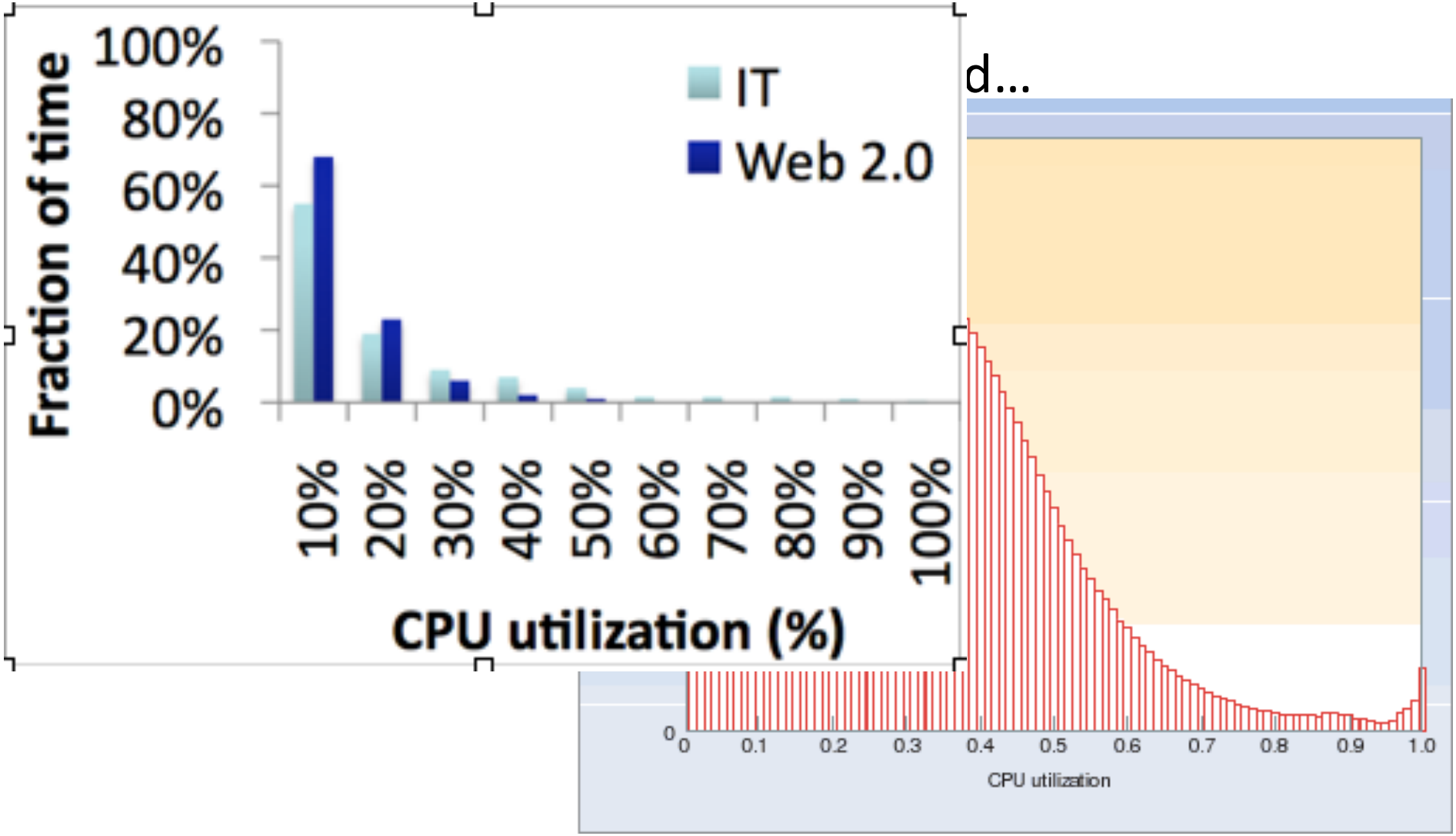
Power Infrastructure



Power Infrastructure

- Solutions:
  - Avoid unnecessary transformations on motherboard; use high-quality parts
  - Avoid transformations & UPS
  - Increase efficiency of conversions
  - Bring high voltage close to machines
  - Others
- In particular:
  - Avoid generator/battery spending by using many smaller distributed datacenters

Utilization

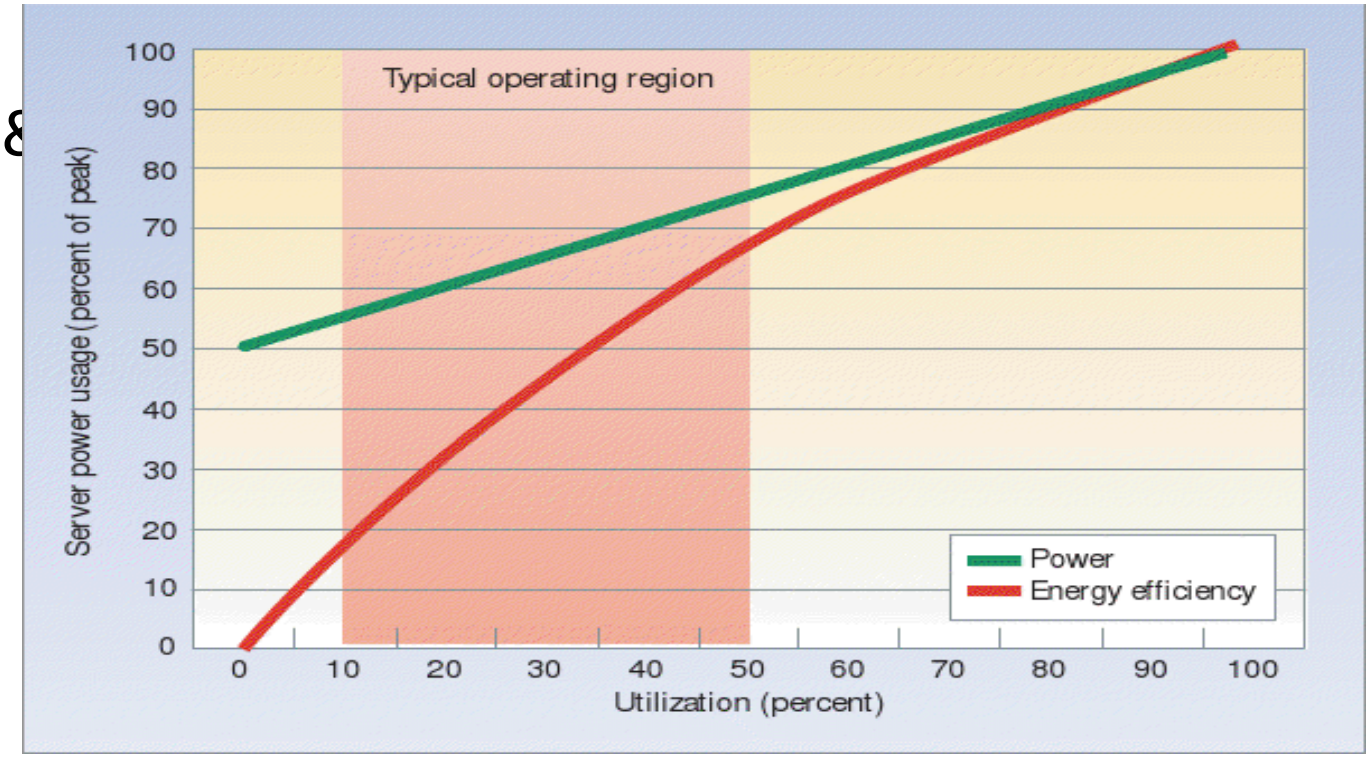


Utilization

- Why so poorly utilized?
  - Work spread over many machines for robustness, data safety
  - Natural variance in load means most times will not be peak
    - Is it ever possible to do more work during off-peak times to reduce work during peak times?
    - Often, no
  - Server-class machines often mismatched to Web workloads
  - Many background tasks mean machines never completely idle

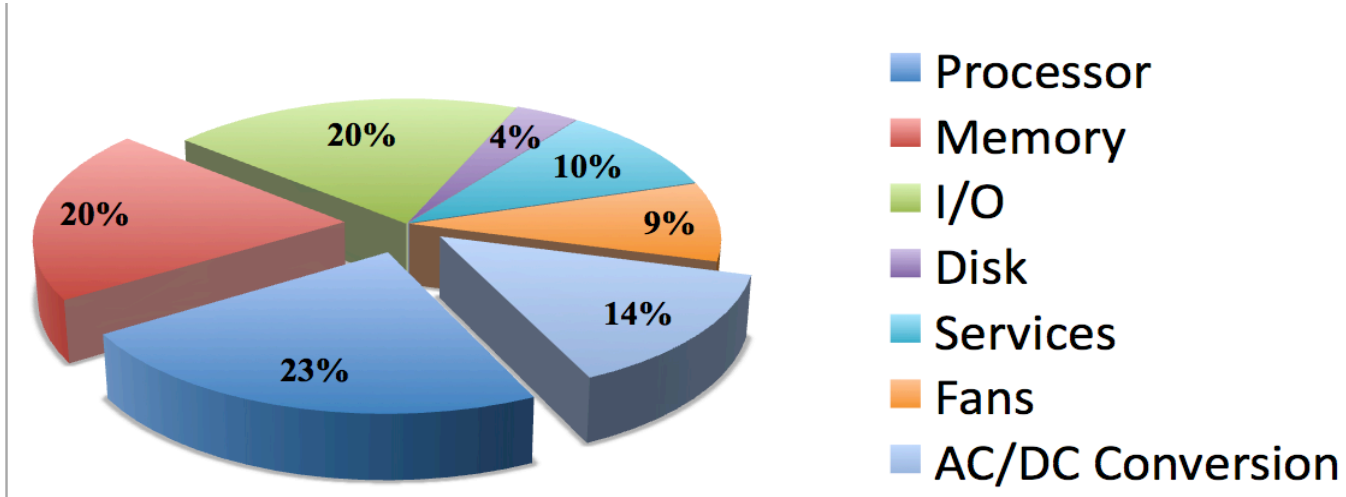
Energy-Proportionality

- … &



- Why?

Energy-Proportionality



- No one component dominates
- CPU has better dynamic range than others, esp. on embedded processors
- Other components only Off or On
  - Or for networking, just On!

Energy-Proportionality

- Solutions:
  - Better dynamic range from DRAM, disks, network
  - Build systems that are low peak-perf but high-throughput per $. Uses embedded & slower processors.

Sharing is Good

- If done well, can greatly reduce costs for all
- If done poorly, can cause contention at hot spots
- In real-life:
  - Time share vacation rentals
  - Library books
  - …
- Someone has to manage the sharing

Sharing Computers and Network

- Standard operating systems today share computer resources across multiple processes (and multiple users)
  - Each process has illusion it alone has the machine
  - Well, kind of

The Cloud

- Rent what you need when you need it
- Can be a lot cheaper
  - Particularly if your demand has high peaks
  - But even if not, due to economies of scale
- But need guarantees of service with provider
  - Service Level Agreements

Do You Trust your Service Provider?

- If you put all your business data in the cloud, someone else is now responsible for it
  - What if they are malicious? Paid off by the competition?
  - What if they are incompetent? How can you catch bugs in their software?
  - What if they go bankrupt? Can you recover your data?
- All addressable concerns, if planned for

A Service:

- Is a logical representation of a repeatable business activity that has a specified outcome (e.g., check customer credit, provide weather data, consolidate drilling reports)
- Is self-contained
- May be composed of other services
- Is a “black box” to consumers of the service

Service vs. API/Object Interface

- All of these are motivated by modularity
- Critical distinguishing feature of service is loose coupling
  - Different process
  - Usually different machine
  - Usually managed by another party
- Invoke service and obtain results through messages.
- Service providers usually guarantee (some) characteristics of the service

Service Oriented Architecture

- Service-orientation is a way of thinking in terms of services and service-based development and the outcomes of services
- SOA is a paradigm for organizing and utilizing distributed capabilities that may be under the control of different ownership domains.
- AWS grew out of Amazon's internal SOA efforts

Interact with Services w/ REST

- GET used to retrieve items from DB
- PUT used to update items in DB
  - Same URL identification of item to update
  - PUT body specifies actual update to apply.
- POST used to insert items into DB
  - URL identifies parent of item inserted
  - Body has content of item
  - Server inserts new item; returns its URL
- DELETE used to delete items from DB