Living in a Layered World

Storage and I/O resarch of Andrea Arpaci-Dusseau, Remzi Arpaci-Dusseau, and Miron Livny

of the

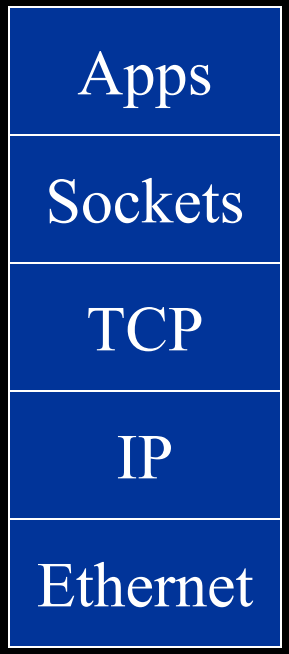
University of Wisconsin, Madison

(as told to John Bent (formerly from Wisconsin (now at

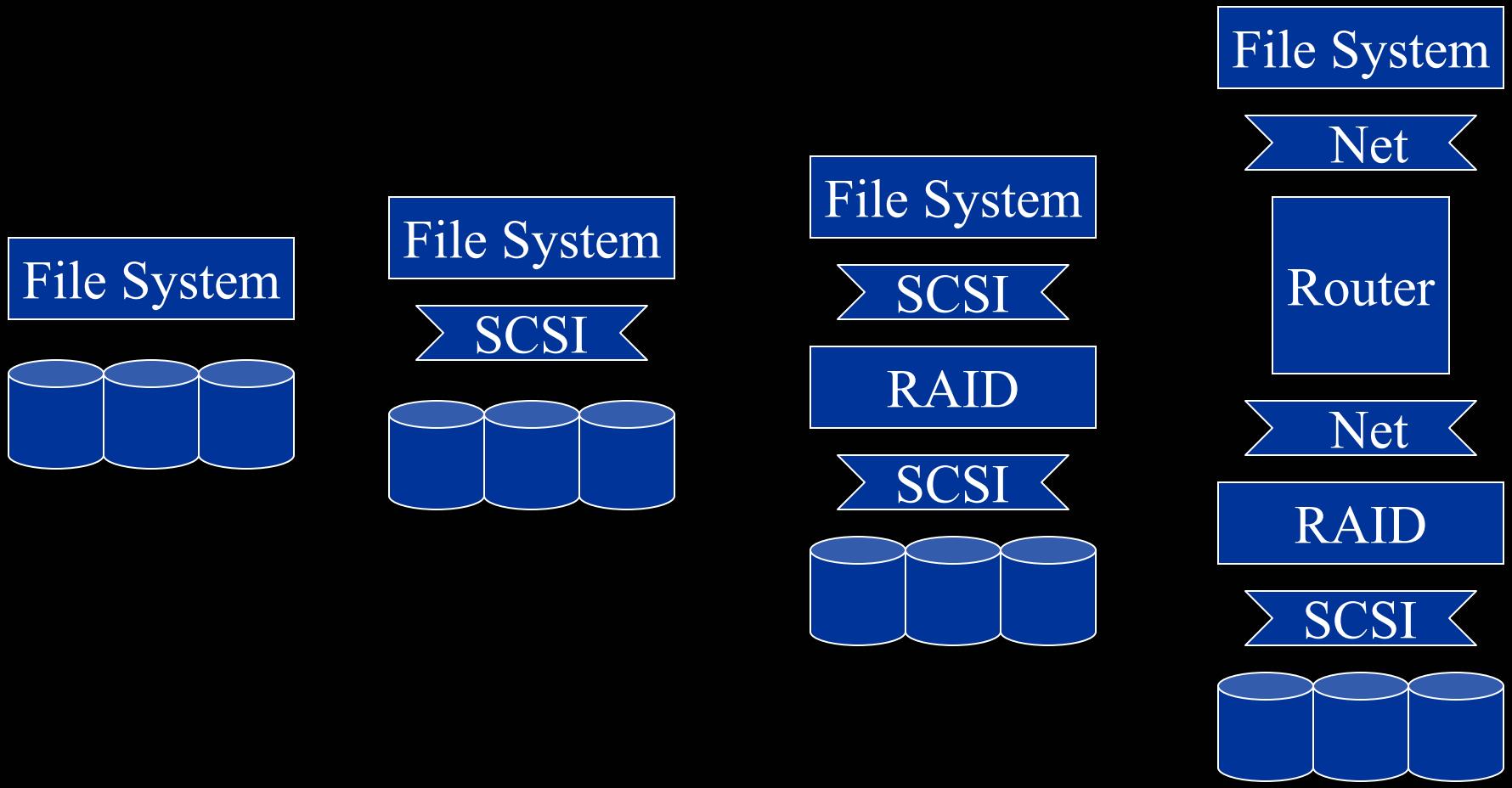
Systems are Built from Layers

Layers are good:

- Decrease in complexity
- Increase in modularity



Storage: Increasingly Layered



Too Much of a Good Thing?

Layers are bad:

- Loss of performance
- Loss of information
- Loss of perspective

Hypothesis One: Loss of info prevents us from building many new and interesting systems

- Our analysis research: Techniques to cope with information loss
  (how to learn about and exploit other layers)

Hypothesis Two: Loss of perspective leads to over-emphasis within a single layer

- Our end-to-end reseach: Complete systems designed to explicity work across entire stack of layers

Outline

- Analysis: Bottom up
  - Semantically-smart disks
    - D-GRAID
    - Secure delete
  - Better second-level caching with X-RAY
- Analysis: Top down
  - Deducing RAID properties: Shear
- System analysis: Coping with complexity
  - EMC Centera
- End-to-end systems
  - IRON-FS
  - BAD-FS and Stork

Block-based Storage: Too Dumb

The Problem is the Interface: Too Narrow

But, You Can’t Change the Interface:
Joy’s Law

What to Do?

Semantically-Smart Disk System (SDS)

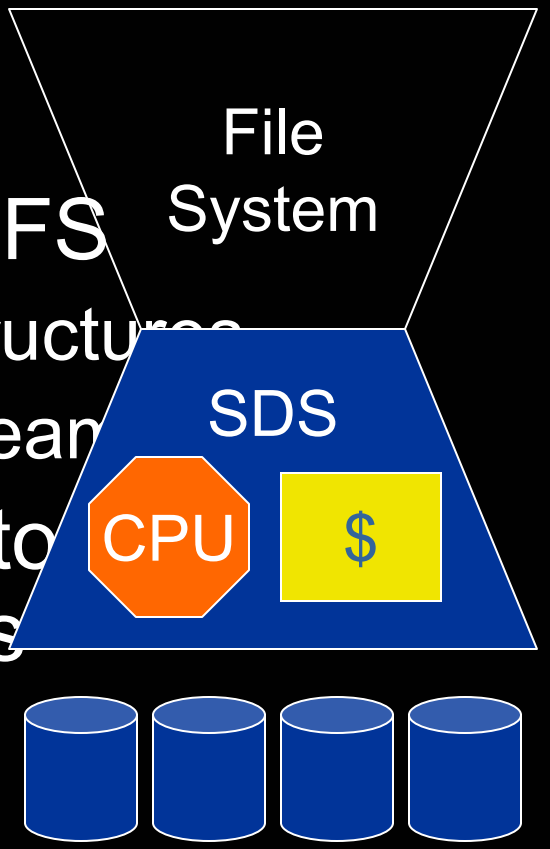
Disk system that understands file system

- Data structures
- Operations

Operates underneath unmodified FS

- Must discover layout + on-disk structures
- Must "reverse engineer" block stream

Exploits knowledge and "smarts" to implement new class of services

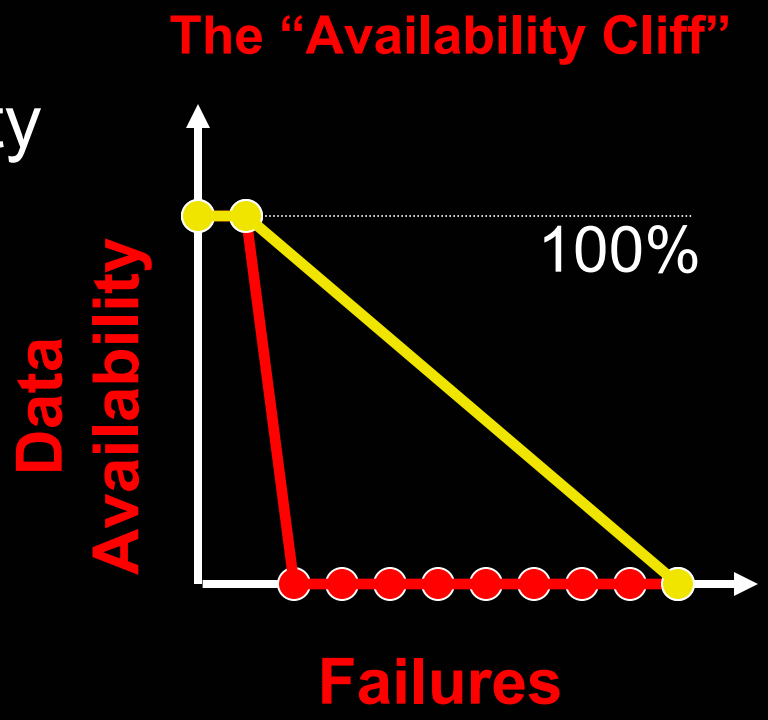


Improving Storage availability with D-GRAID

Paramount to system availability
- Downtime → millions of $ /hr

Current solution: RAID
- Handles 1 failure well

Problem
- Handles multiple failures poorly



Answer: D-GRAID uses semantic info to gracefully degra

Secure Delete

Data block contents live on - long after deletion

- Could be undesirable

Idea: Upon delete, “scrub” data blocks

- Overwrite in such a way as to make unrecoverab
- Write-buffering systems can “out-smart” themsel

Semantically-smart disk version

- Must discover when blocks have been deleted

X-Ray: The Levels of Caching

Caching in modern systems

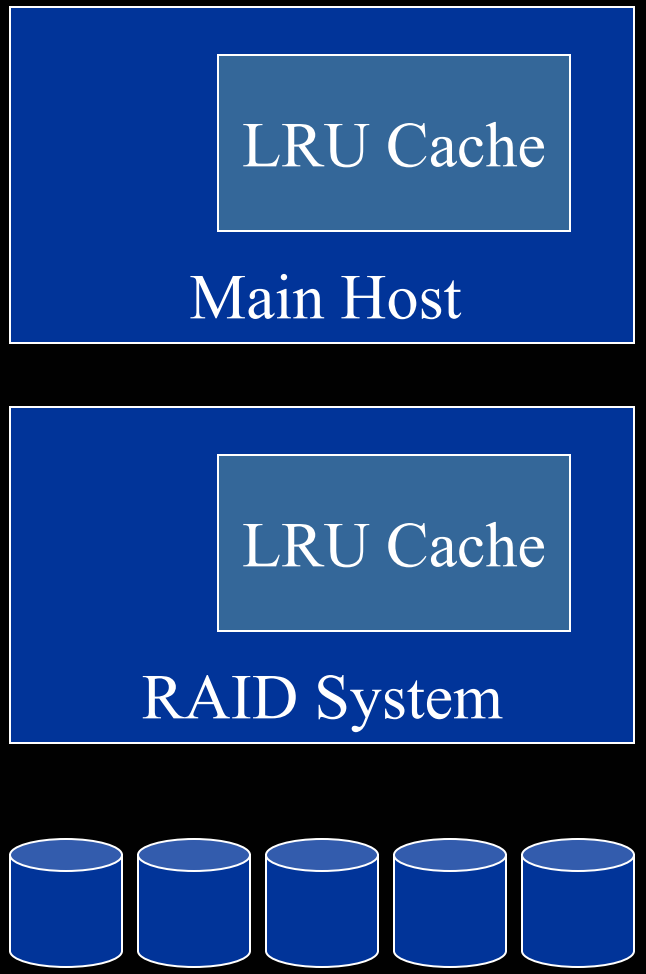
- Multiple levels
- Storage: 2-level hierarchy

Level 1: File system (FS) cache

- Software-managed
- Main memory of host/client
- LRU-like cache replacement

Level 2: RAID cache

- Firmware-managed
- Memory inside RAID system
- Usually LRU replacement



The Problem?

The Semantic Solution: X-RAY

The problem: 2nd cache needs to know:

- Contents of 1st cache, to avoid redundant cachin
- Ordering of LRU queue of 1st cache, to cache next most MRU blocks in its cache

But, these things are hard to know!

- Why? Unobservable I/O traffic

Semantic information used:

- Access time updates in inode

Analysis Looking Down
Shear: RAID Deconstruction

Shear: An information-gathering tool

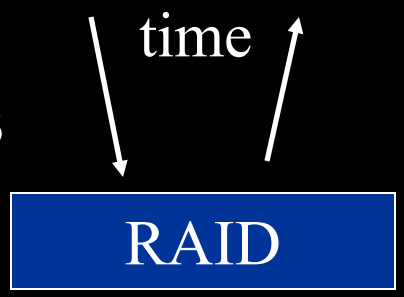
- Runs series of controlled tests against RAID arra

Determines automatically:

- Pattern size, Chunk size, Number of disks, RAID

Key techniques:

- Well-crafted I/O microbenchmarks
- Statistical clustering

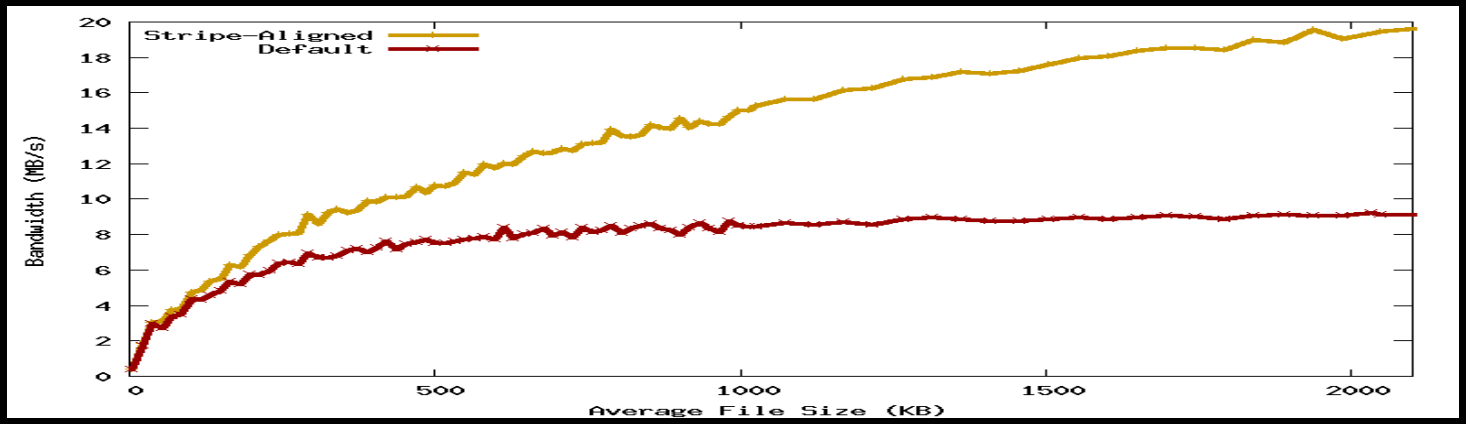


Using Shear Info: Stripe-aligned Writes

Overcome RAID-5 small write problem

Modified Linux disk scheduler

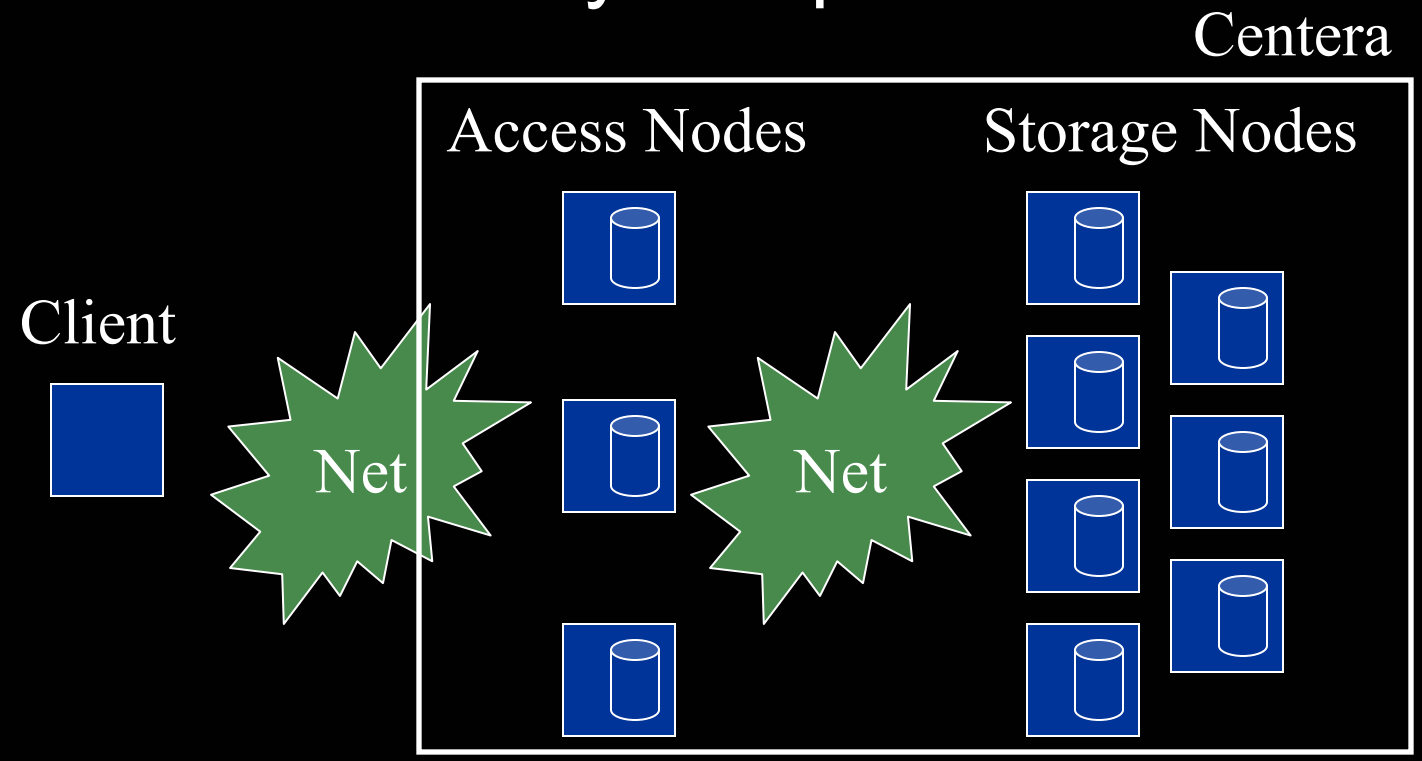
- Groups writes into full stripes
- Aligns writes along stripe boundaries
- Approximately 20 lines of code



Intra-system analysis: EMC Centera

Purpose: Archival storage

Built from commodity components



Intra-box Techniques

Two “Intra-box” techniques

- **Observation**
- **System perturbation**

Two components of analysis

- Deduce **structure** of main communication protoco
  - Object Read and Write protocol
- Internal **policy** decisions
  - Caching, prefetching, write buffering, load balancing, et

Summary of findings

Write Policies	
Replication	Two copies in two nodes attached to different power (reliability)
Load balancing	CPU usage (locally observable status) Network status is not incorporated
Write buffering	Storage nodes write synchronously

Read Policies	
Caching	Storage node only (commodity filesystem) Access node and client does not cache.
Prefetching	Storage node only (commodity filesystem) Access node and client does not prefetch
Load Balancing	Not implemented in earlier version Still reads from busy nodes

Outline

- Analysis: Bottom up
  - Semantically-smart disks
    - D-GRAID
    - Secure delete
  - Better second-level caching with X-RAY
- Analysis: Top down
  - Deducing RAID properties: Shear
- System analysis: Coping with complexity
  - EMC Centera
- End-to-end systems
  - IRON-FS
  - BAD-FS and Stork

Storage Trend: More Problems

Denser drives: Capacity sells drives

- More logic -> more complexity
- More complexity -> more bugs

Cost per byte dominates: “Pennies matter”

- Manufacturers will cut corners
- Reliability features are the first to go

Increasing amount of software:

- ~400K lines of code in modern Seagate drive
- Hard to write, hard to debug

Finally, fail-stop model is incomplete

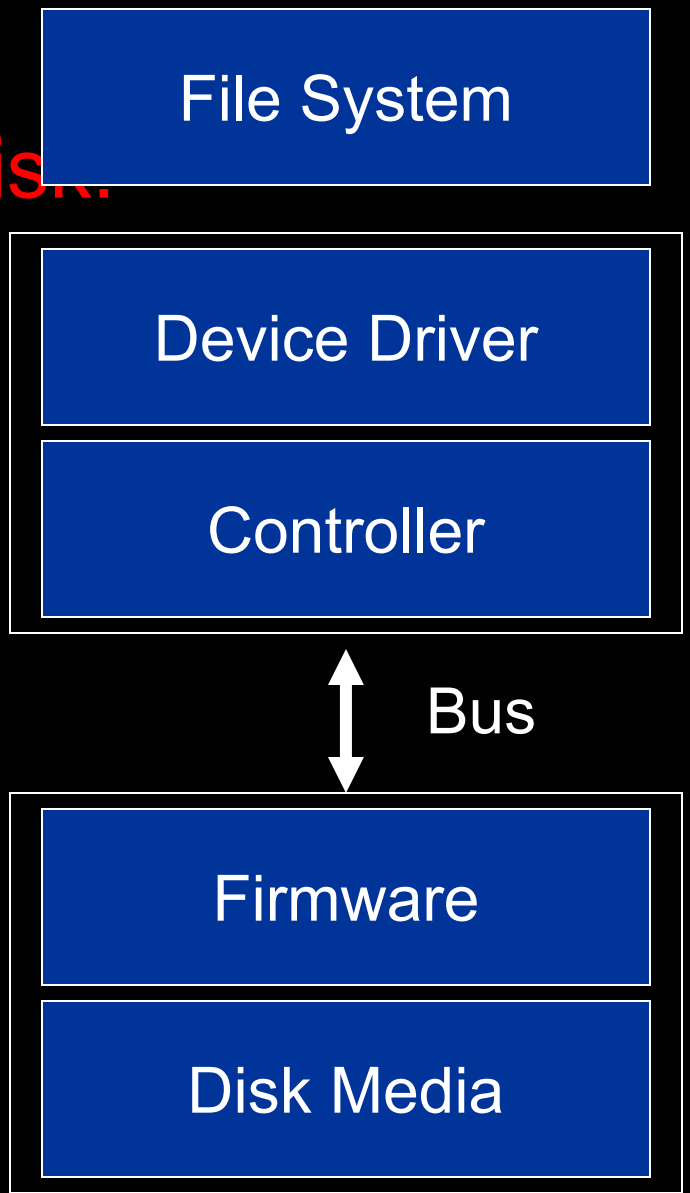
What Should We Do About It?

Paranoid File Systems (IRON-FS)

File System: Don't trust the disk.

Highest level in I/O stack must verify correctness of storage operations

Refined view: Don't trust anything in the I/O stack (Reagan: "Trust but verify")



And now for something completely different: BAD-FS and Stork

Wide-area batch computing has long been CPU-centric

Data-intensive applications suffer

Previous (intra-layer) approaches

- Optimize data movement after CPU-centric schedule is effected

A (inter-layer) end-to-end approach

- BAD-FS and Stork use data considerations to schedule
- Requires coordination between data transfer

A Layered World

Layers exist

- Fact of life that is only getting worse

What we need

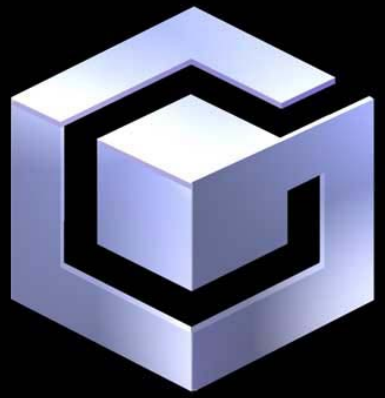
- Tools and techniques to cope with layers
  - Enables us to build more functional and interesting systems
- Systems designed for end-to-end

www.cs.wisc.edu/adsl

www.cs.wisc.edu/condor

Students (People who do the work)

- Nitin Agrawal
- Lakshmi Bairavasundaram
- John Bent (now at LANL)
- Nate Burnett
- Tim Denehy
- Haryadi Gunawi
- Todd Jones
- Ina Popovici
- Vijayan Prabhakaran
- Muthian Sivathanu (now at Google)
- Doug Thain (now at Notre Dame)



Professors

- Andrea Arpaci-Dusse
- Remzi Arpaci-Dussea
- Miron Livny