

Disconnected Operation in the Coda File System

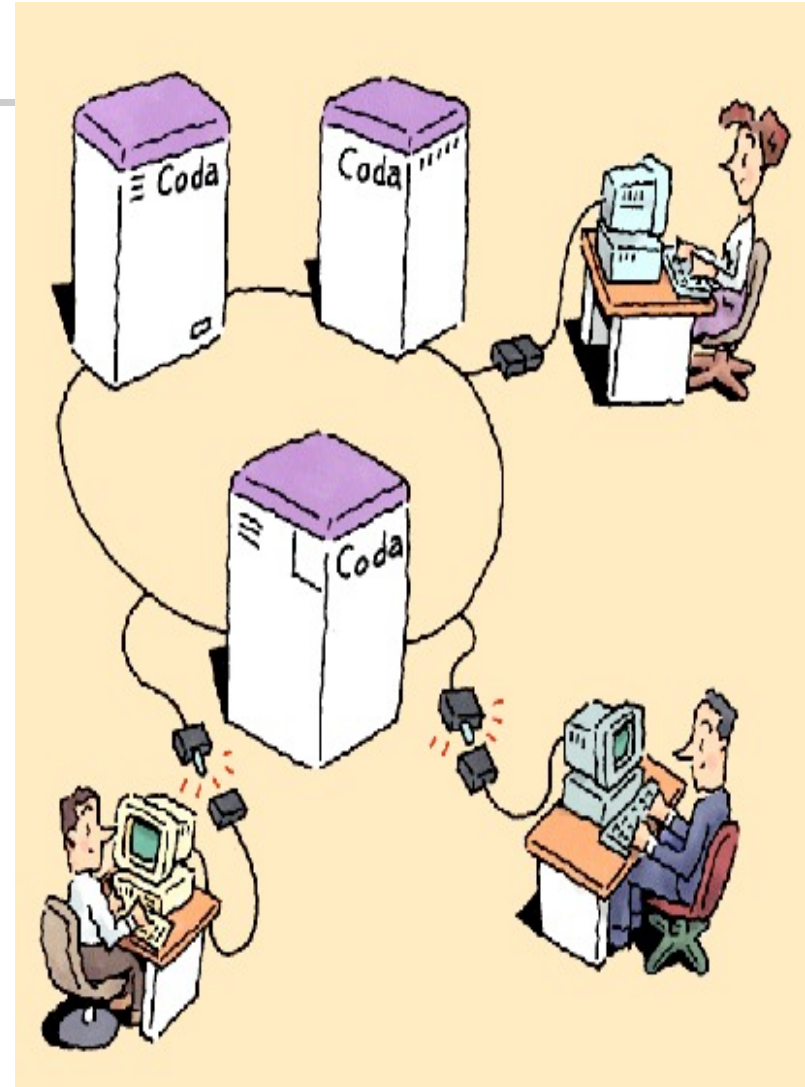
James J. Kistler
M. Satyanarayanan

Background

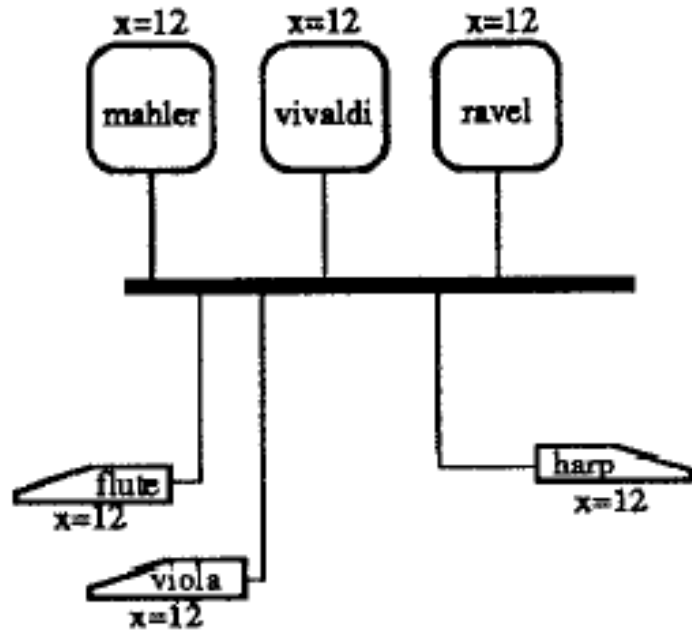
- Distributed file system
 - Store files on one or more servers accessible by clients
 - Advantages: large capacity, data sharing, application sharing, backup, manageability ...
 - Challenges:
 - Security
 - **Remote failures: network failure, server failures**
- Workstations are powerful!

Coda File System

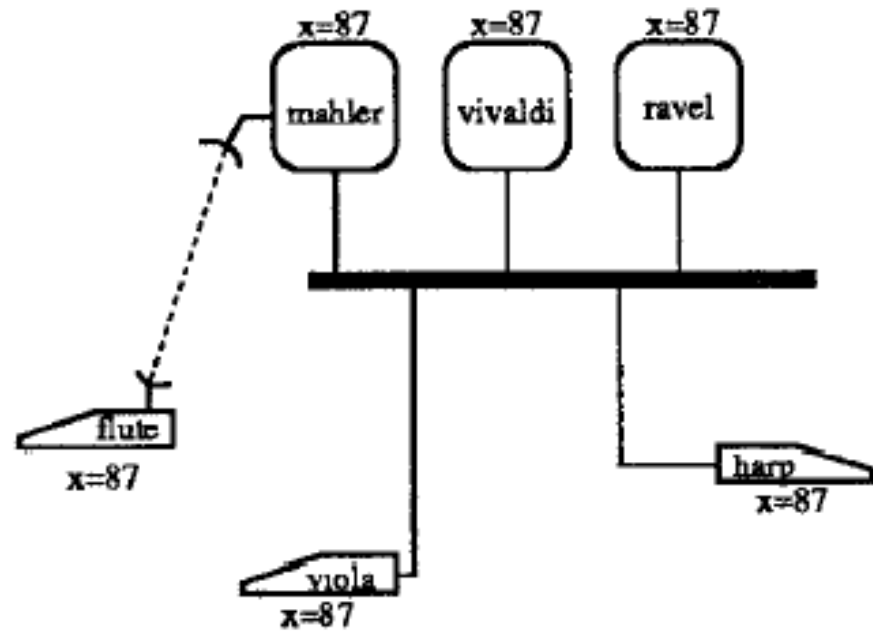
- *Use caching of data for availability as well as performance*
- Enjoy the benefits from both distributed FS and “powerful” workstations



Disconnected operation(1)

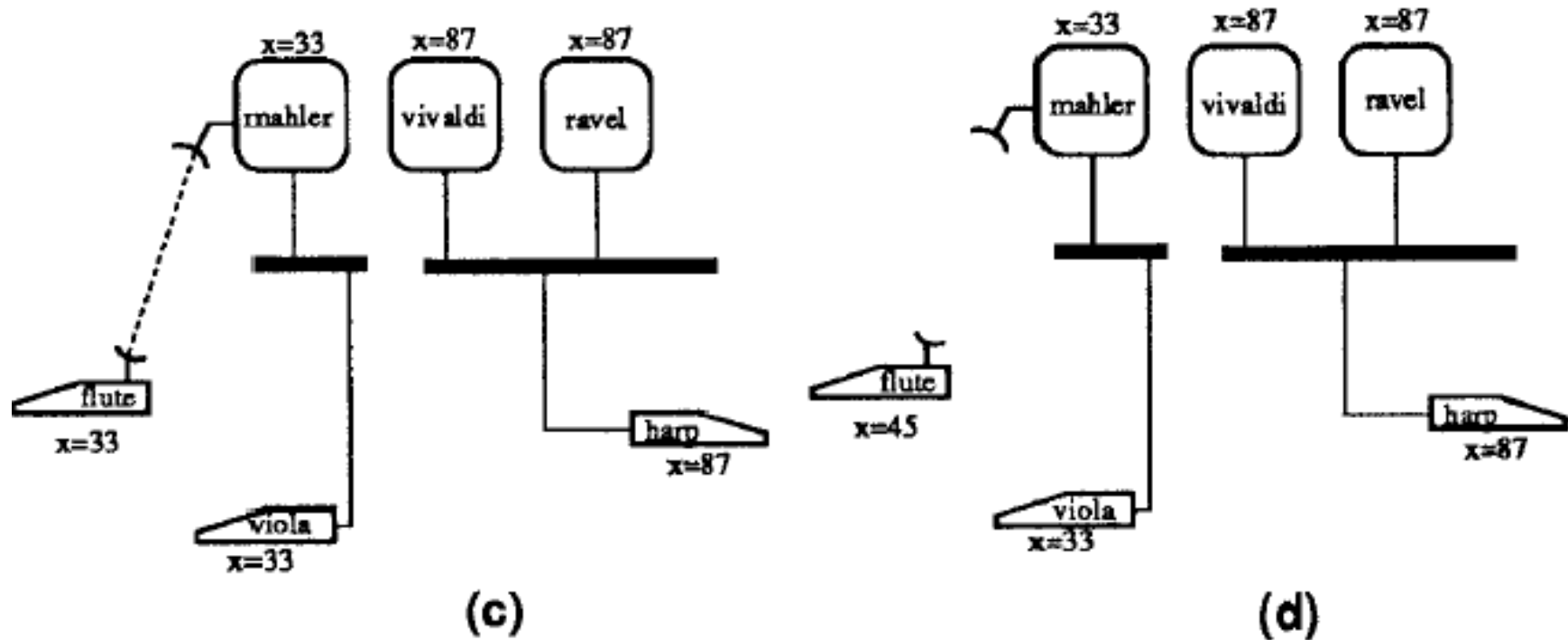


(a)

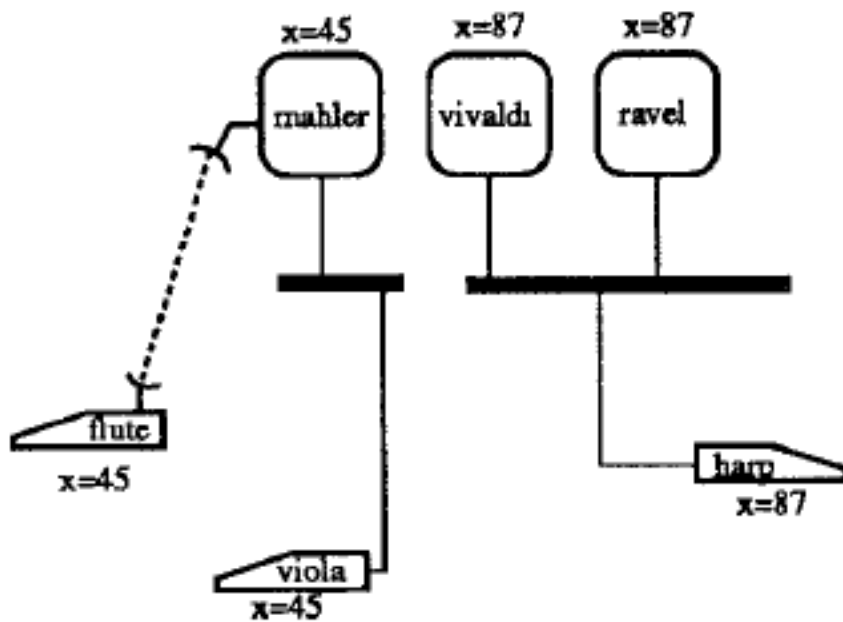


(b)

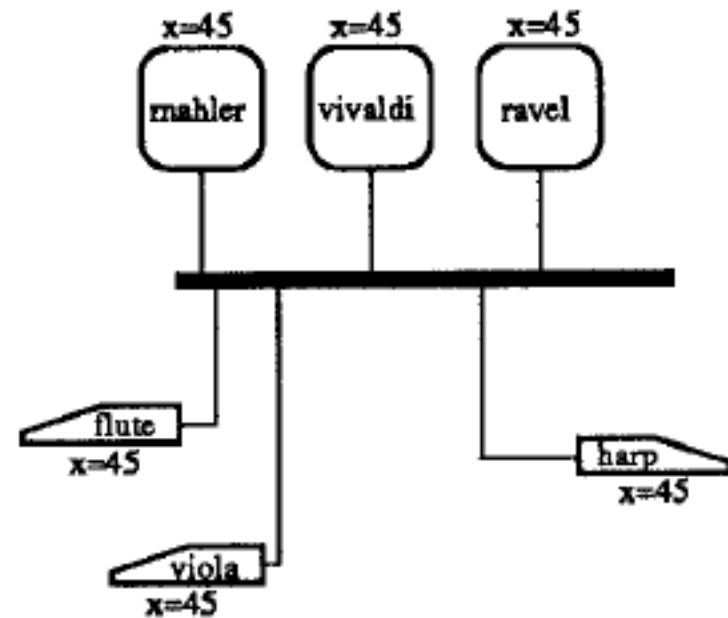
Disconnected operation(2)



Disconnected operation(3)



(e)



(f)

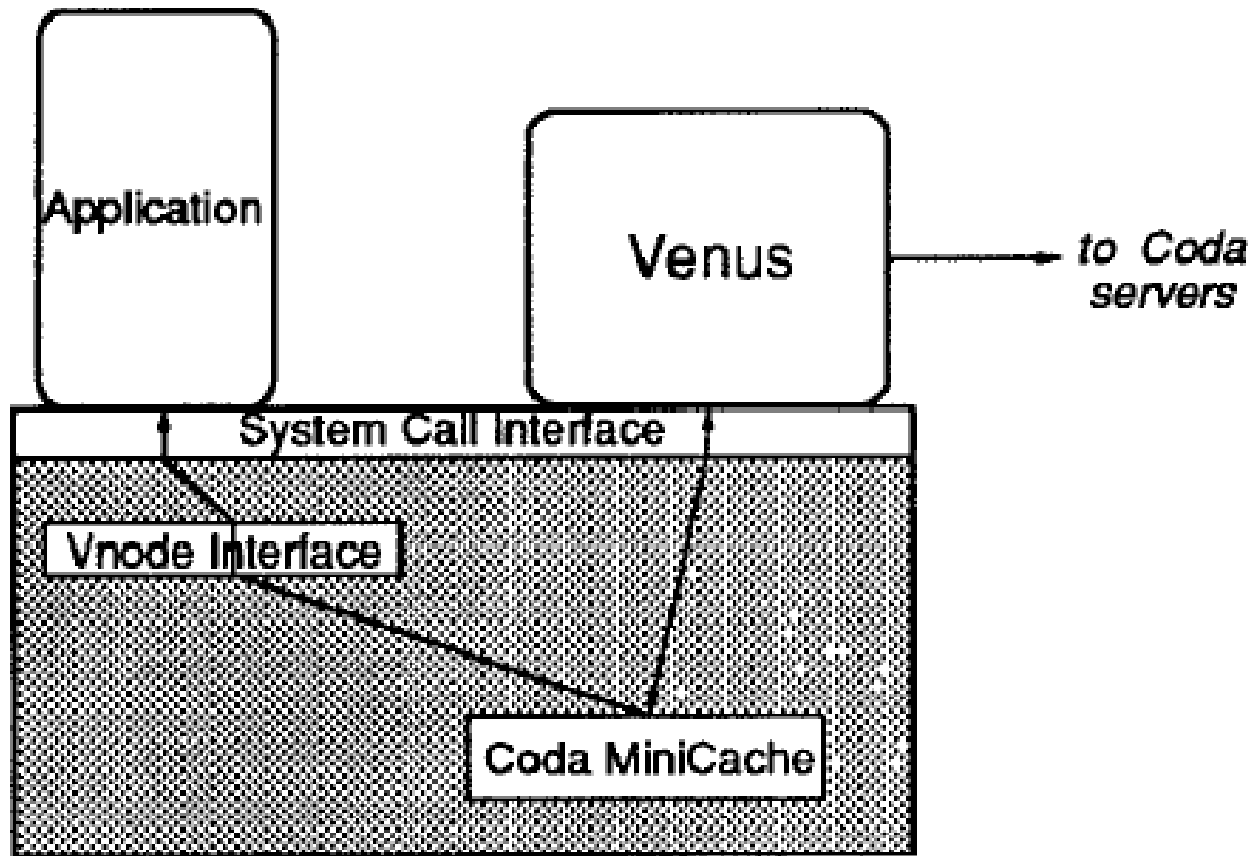
Design Rationale of Coda

- Scalability
 - Callback based cache coherence
 - Whole-file caching
 - Fat client
 - Slow system-wide change
- Portable Workstations
 - user assistance

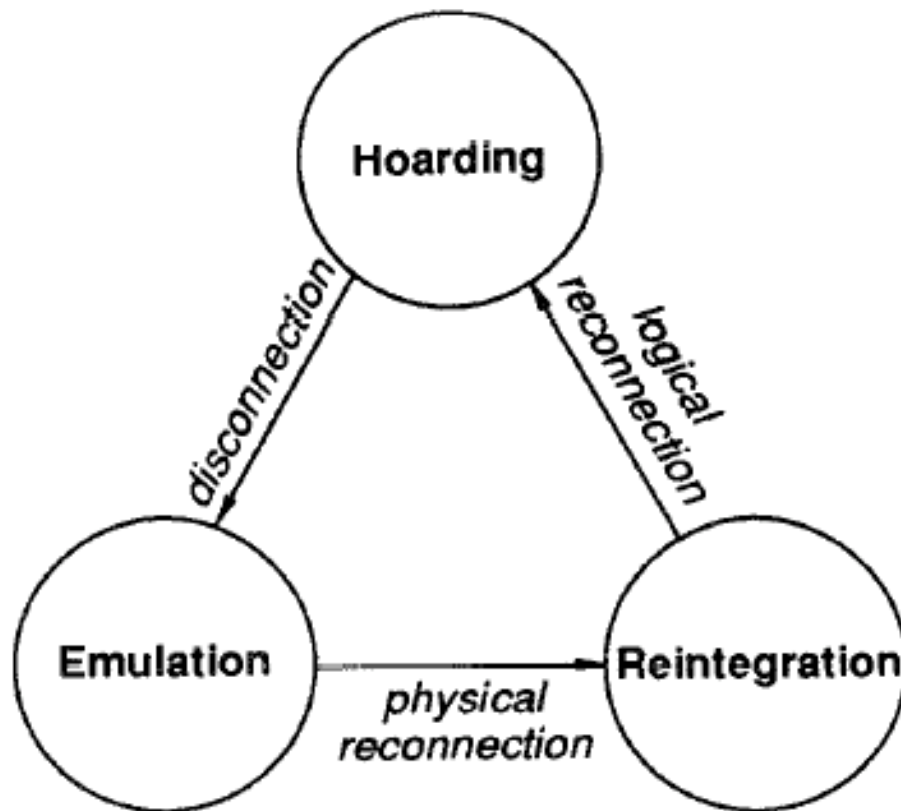
Design Rationale of Coda (cont)

- First vs. Second Class Replication
 - First class: server replication
 - Capacity, Physical security; Expensive
 - Second class: client caches
 - Performance, Scalability, Cost; Quality
- Optimistic vs. Pessimistic Replica control
 - Pessimistic: disallowing all writes on disconnection, or only R/W on one partition
 - Optimistic: allow R/W anywhere
 - *low degree of write-sharing typical of Unix*

Implementation: client structure



Venus: state machine



Venus: Hoarding

- What's the purpose?
 - Hoard important data
 - Hoard extra permanent fids
- What's "important data"?
 - Prioritized Cache Management
 - Per-workstation Hoard Database (HDB)
 - User assistance
- Hoard Walking
 - Issue: higher priority data might not in cache:
 - Replaced by low priority data
 - Invalidated by callback cache coherence process
 - Solution: "hoard walk" every 10 mins

Venus: Emulation

- Emulate servers on disconnection
- Per-volume log
- Persistence: Non-volatile storage
 - Metadata: RVM
 - Actual cache & log content: local Unix files
- Resource exhaustion:
 - Issue: no enough storage for replay log
 - Current: delete files or freeze mutation
 - Compress log? User-assisted deletion/backup?

Reintegration

- Obtain permanent fids for new objects
- Replay the log independently at each server:
 - Parse → validate → execute
→ transfer stored data → commit
 - Why done independently at each server ??
 - How if different servers generate different storeIDs?
 - How if replay fails on some of the servers?
- Conflict handling
 - Compare *storeid*: file & directory
 - Write conflicted replay logs back to local disk

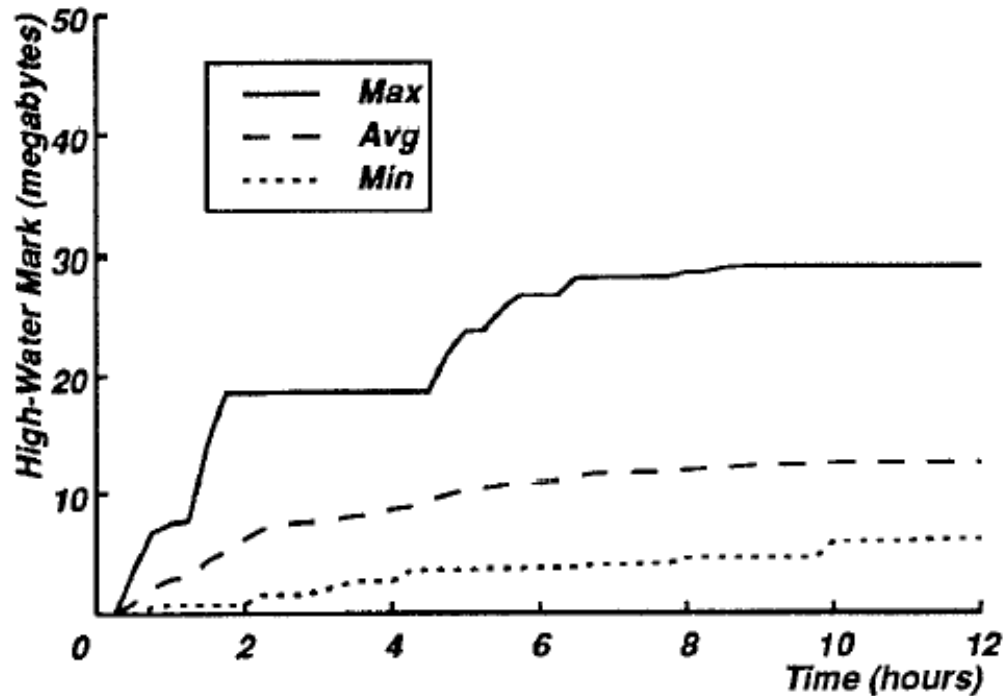
Duration of Reintegration

	Elapsed Time (seconds)	Reintegration Time (seconds)				Size of Replay Log		Data Back-Fetched (Bytes)
		Total	AllocFid	Replay	COP2	Records	Bytes	
Andrew Benchmark	288 (3)	43 (2)	4 (2)	29 (1)	10 (1)	223	65,010	1,141,315
Venus Make	3,271 (28)	52 (4)	1 (0)	40 (1)	10 (3)	193	65,919	2,990,120

This data was obtained with a Toshiba T5200/100 client (12MB memory, 100MB disk) reintegrating over an Ethernet with an IBM RT-APC server (12MB memory, 400MB disk). The values shown above are the means of three trials. Figures in parentheses are standard deviations.

Table 1: Time for Reintegration

Cache Size



- In non-ideal case, what is the miss rate given 30MB, 60MB, or 100MB storage?

Likelihood of conflicts

Type of Volume	Number of Volumes	Type of Object	Total Mutations	Same User	Different User					
					Total	< 1min	< 10 min	< 1hr	< 1 day	< 1 wk
User	529	Files	3,287,135	99.87 %	0.13 %	0.04 %	0.05 %	0.06 %	0.09 %	0.09 %
		Directories	4,132,066	99.80 %	0.20 %	0.04 %	0.07 %	0.10 %	0.15 %	0.16 %
Project	108	Files	4,437,311	99.66 %	0.34 %	0.17 %	0.25 %	0.26 %	0.28 %	0.30 %
		Directories	5,391,224	99.63 %	0.37 %	0.00 %	0.01 %	0.03 %	0.09 %	0.15 %
System	398	Files	5,526,700	99.17 %	0.83 %	0.06 %	0.18 %	0.42 %	0.72 %	0.78 %
		Directories	4,338,507	99.54 %	0.46 %	0.02 %	0.05 %	0.08 %	0.27 %	0.34 %

This data was obtained between June 1990 and May 1991 from the AFS servers in the `cs.cmu.edu` cell. The servers stored a total of about 12GB of data. The column entitled "Same User" gives the percentage of mutations in which the user performing the mutation was the same as the one performing the immediately preceding mutation on the same file or directory. The remaining mutations contribute to the column entitled "Different User".

Table 2: Sequential Write-Sharing in AFS

■ Is this true?

Questions

- In the sample HDB, some system files, like /usr/bin, are given high priority. But they are not like to be changed. Why not deploy them as local file system?
- Is any of today's situation different from early 1990s?
 - In paper: client – 12MB memory, 100MB disk
 - Disk size, network bandwidth, cpu performance increase much faster than disk I/O rate
- Is this good for mobile device which are much less powerful than workstations?