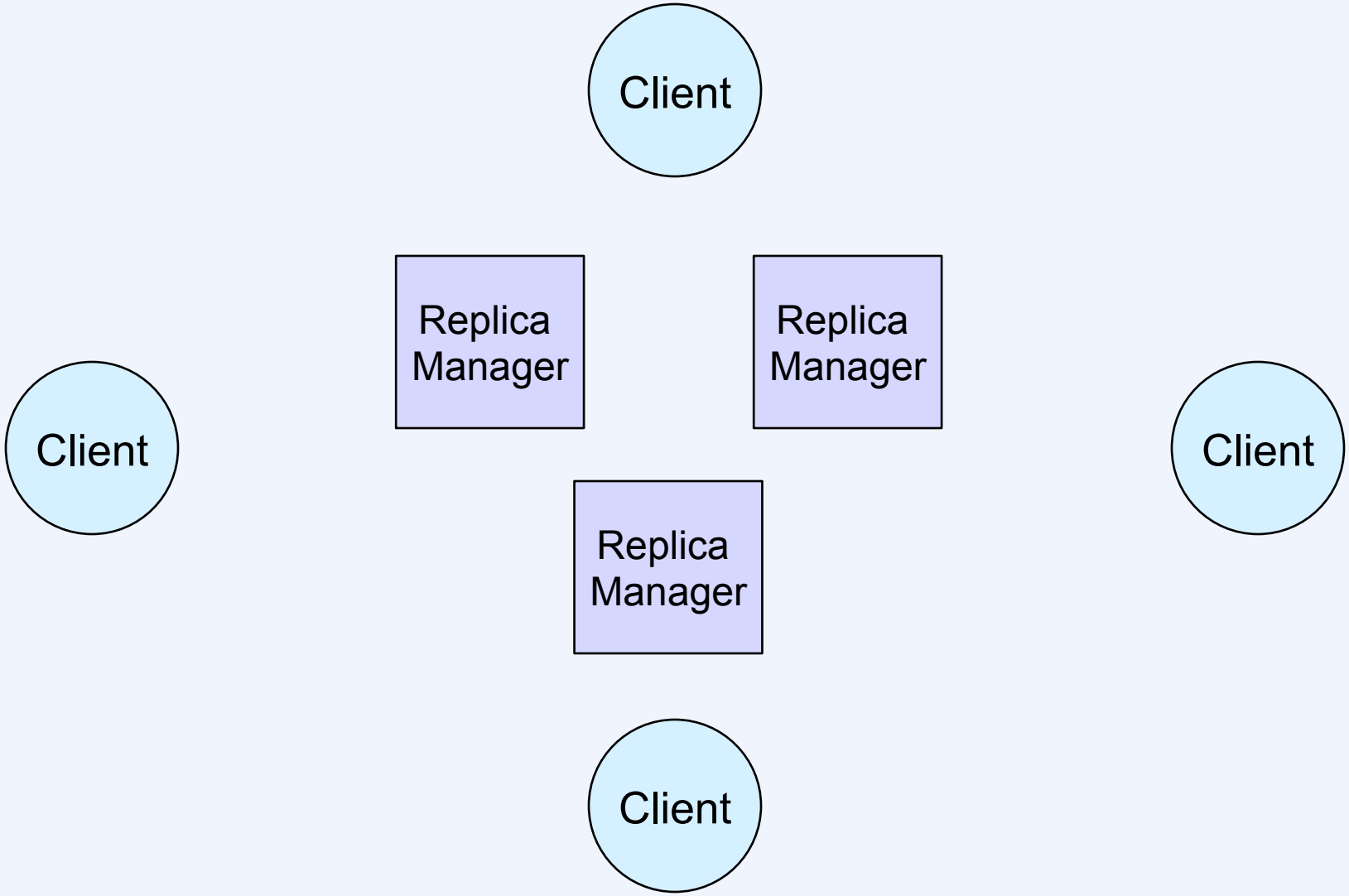
Distributed File Systems (Part 1)

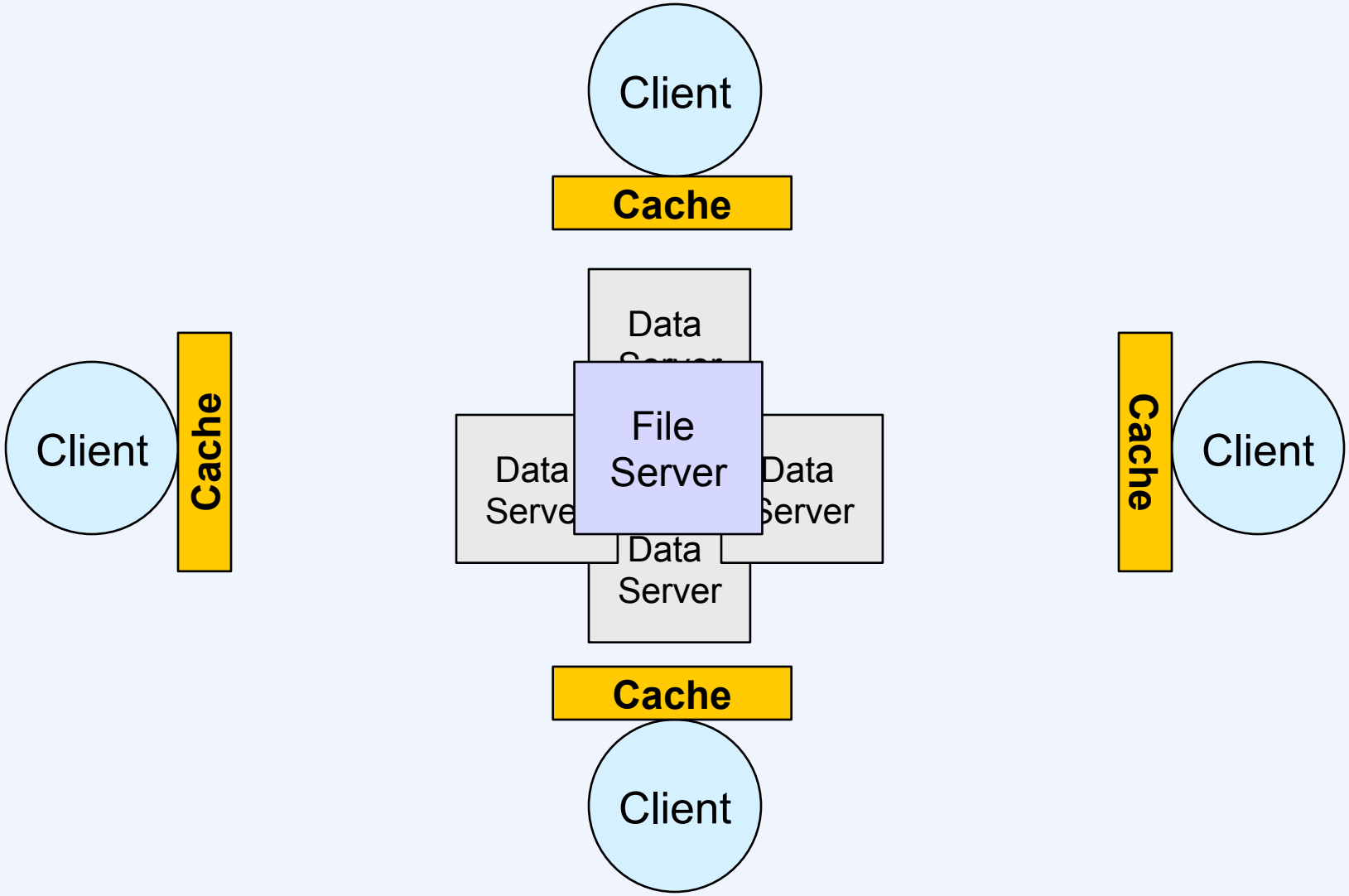
Outline

- Basic concepts
- NFS version 2
- CIFS
- DCE DFS
- NFS version 4
- CS Department: NFS + CIFS + GPFS

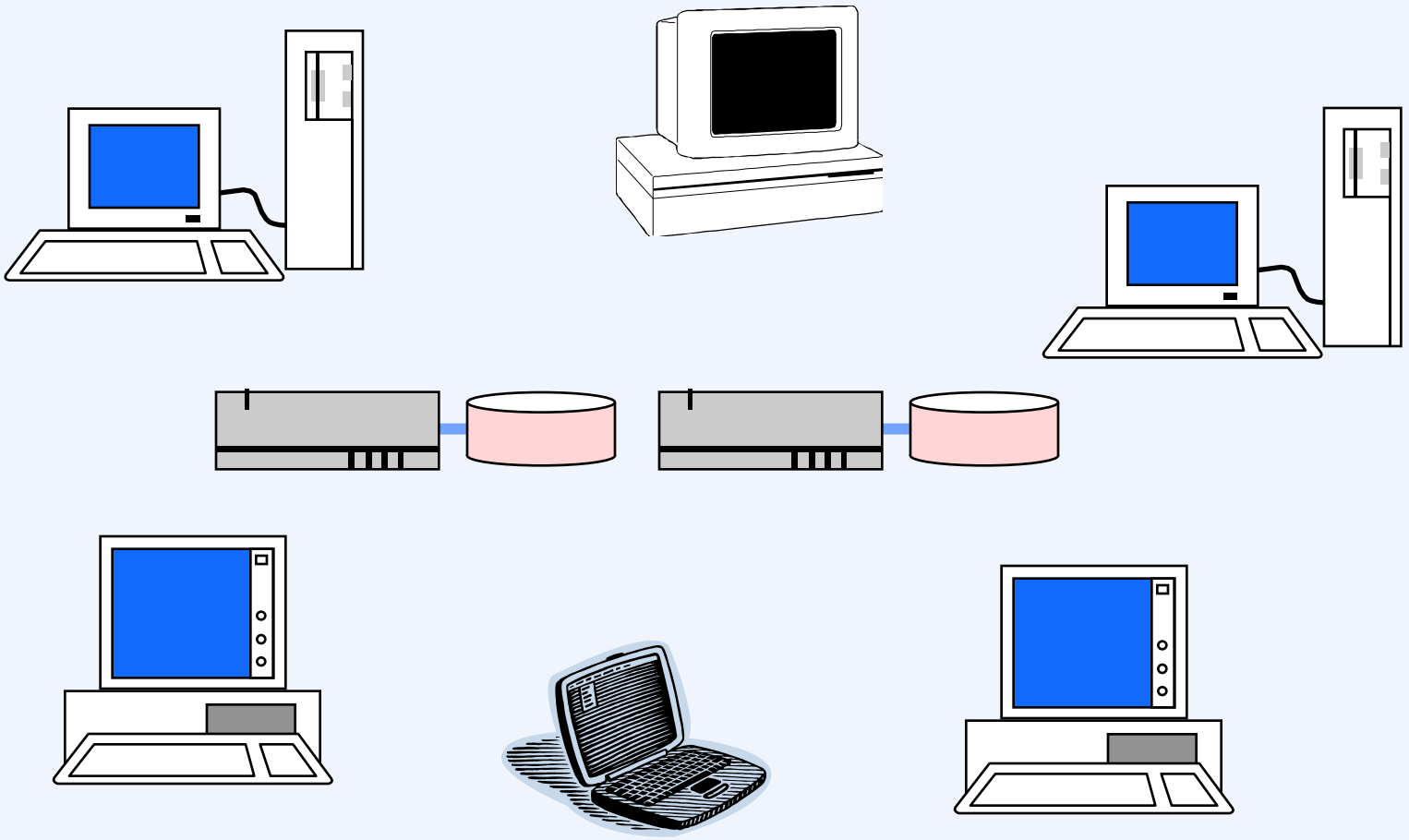
Previous Scenario



DFS Scenario



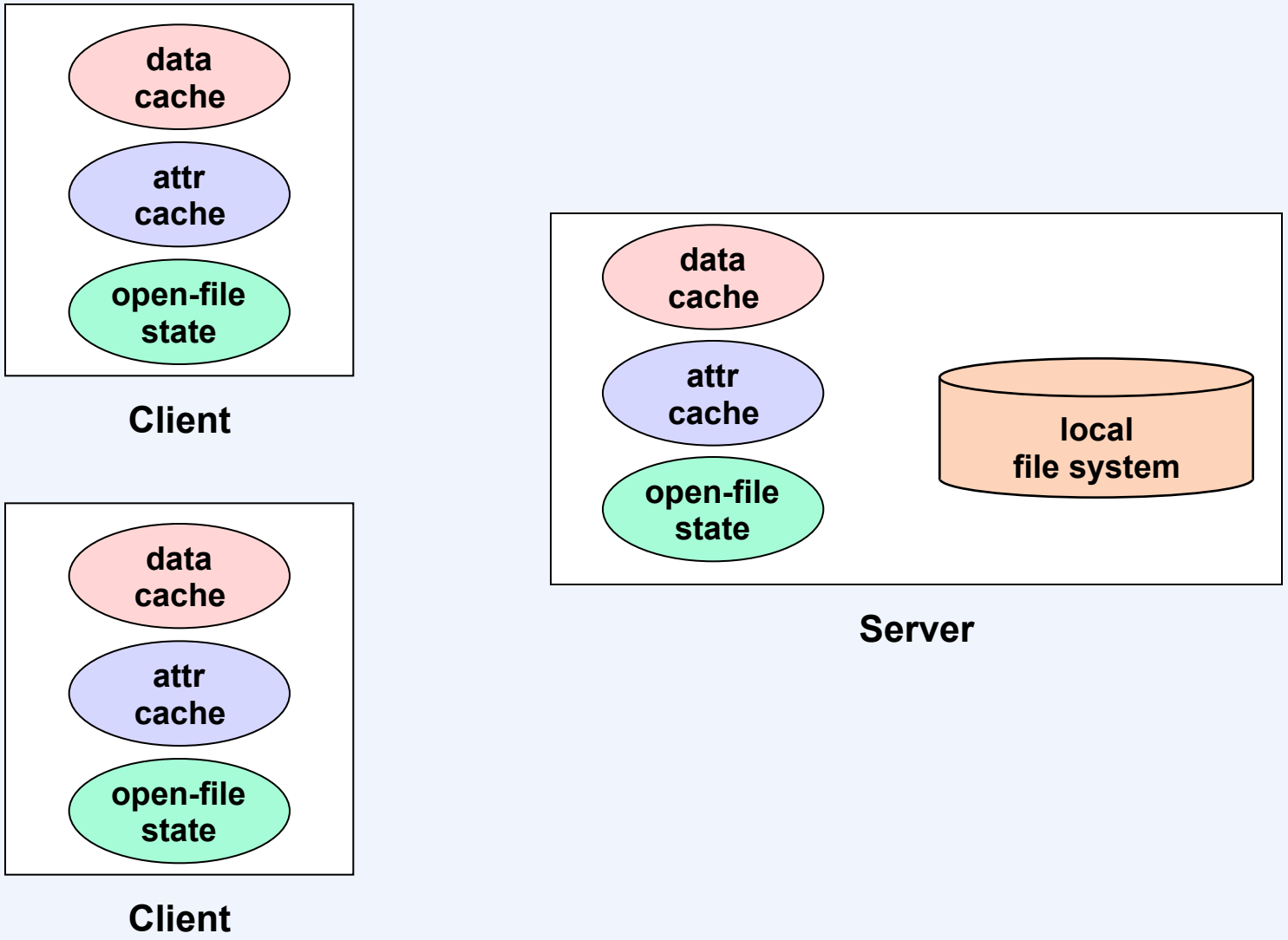
Distributed File Systems



DFS Components

- Data state
  - file contents
- Attribute state
  - size, access-control info, modification time, etc.
- Open-file state
  - which files are in use (open)
  - lock state

Possible Locations



In Practice …

- Data state
  - NFS
    - weakly consistent
    - less weak if program uses locks
  - CIFS and DFS
    - strictly consistent
- Lock state
  - must be strictly consistent

Thursday morning, November 17th
At 7:00 a.m.

Maytag, the department's central file server, will be taken down to kick off a filesystem consistency check.

Linux machines will hang.

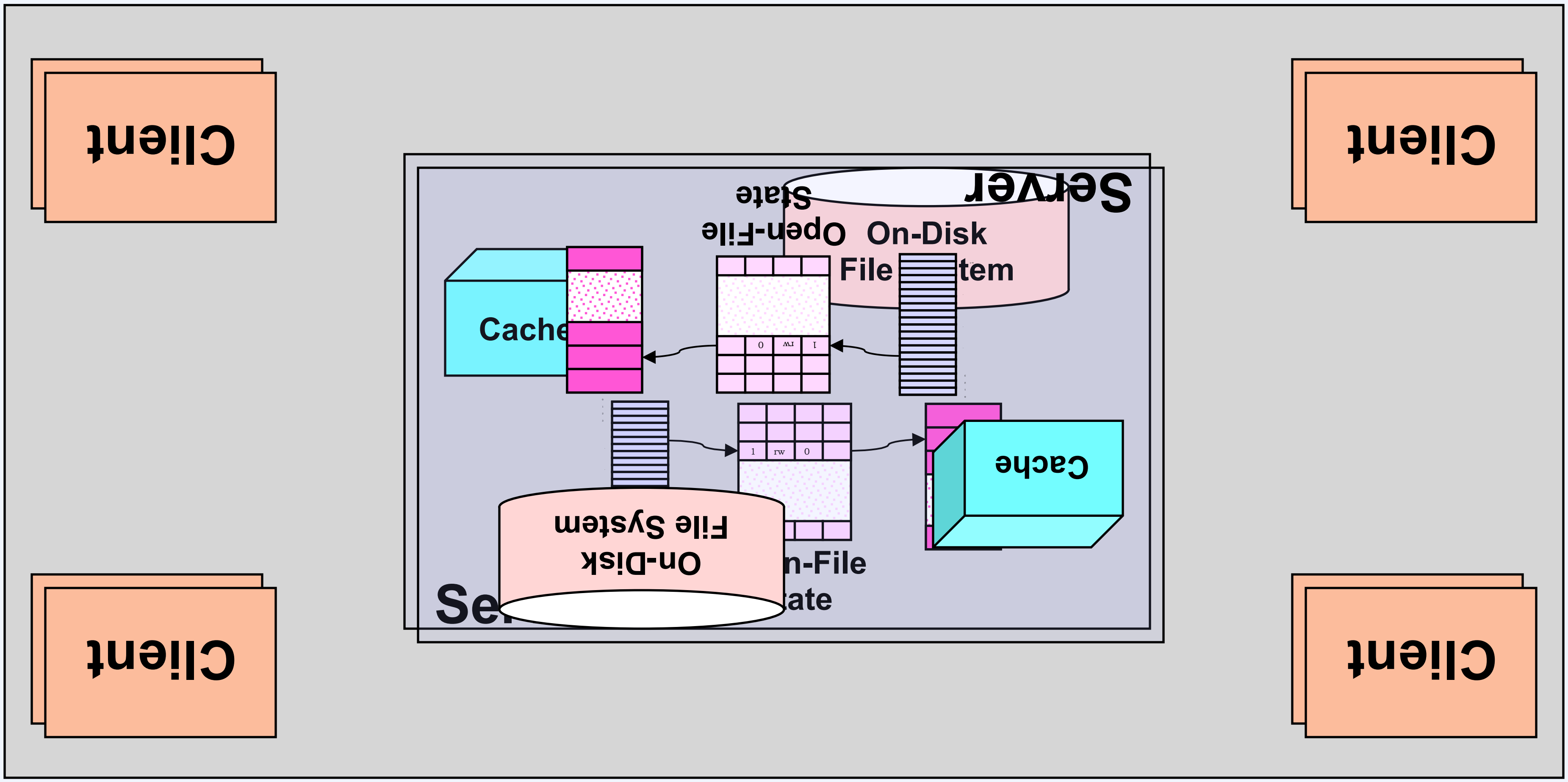
All Windows users should log off.

Normal operation will resume by 8:30 a.m. if all goes well.

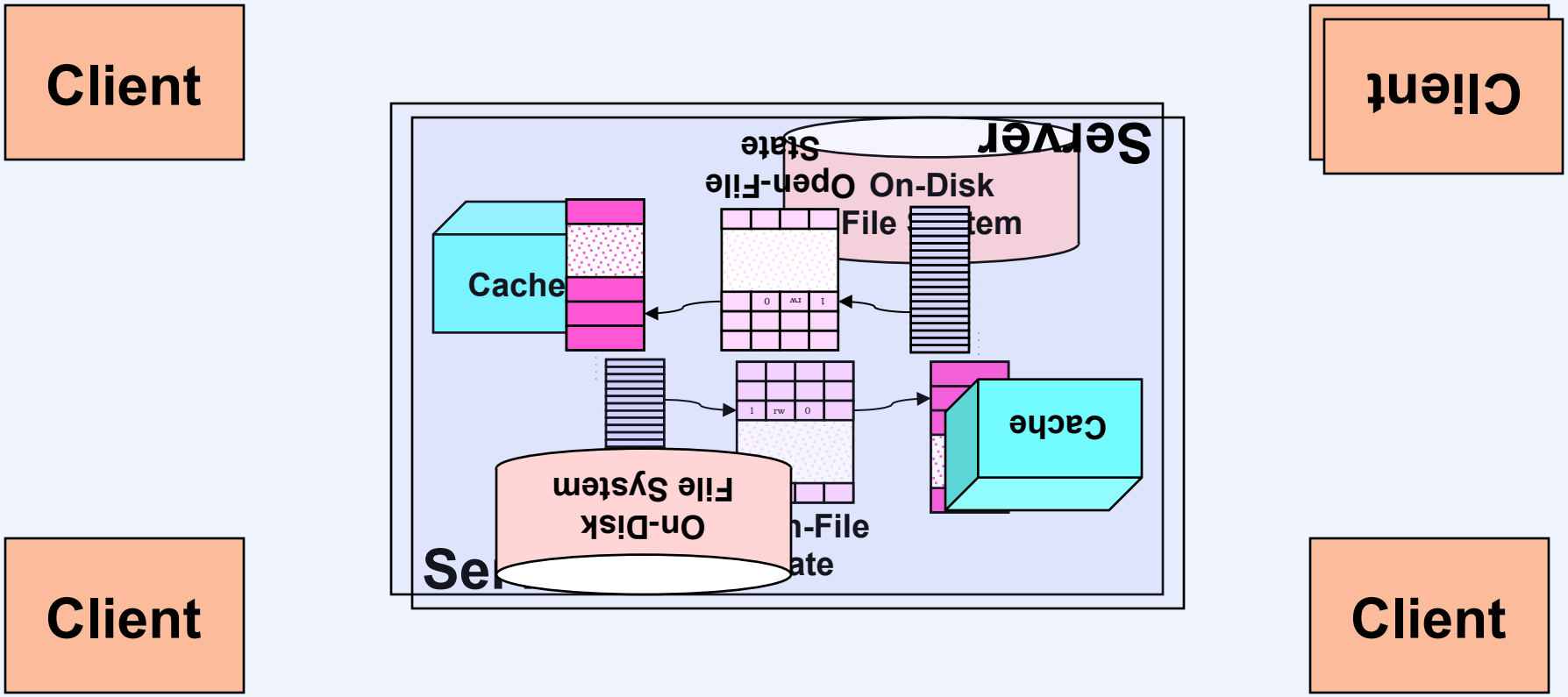
All windows users should log off before this time.

Questions/concerns to problem@cs.brown.edu

Failures in a Local File System



Distributed Failure



In Practice …

- NFS version 2
  - relaxed approach to consistency
  - handles failures well
- CIFS
  - strictly consistent
  - intolerant of failures
- DCE DFS
  - strictly consistent
  - sort of tolerant of failures
- NFS version 4
  - either relaxed or strictly-consistent
  - handles failures very well

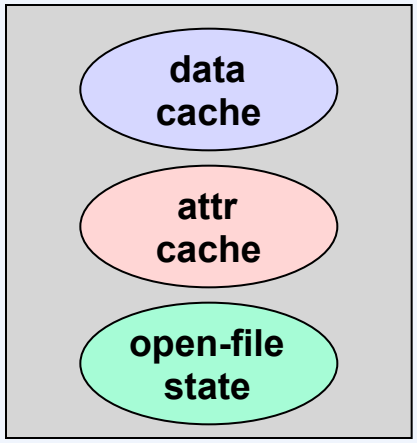
NFS Version 2

- Released in mid 1980s
- Three protocols in one
  - file protocol
  - mount protocol
  - network lock manager protocol

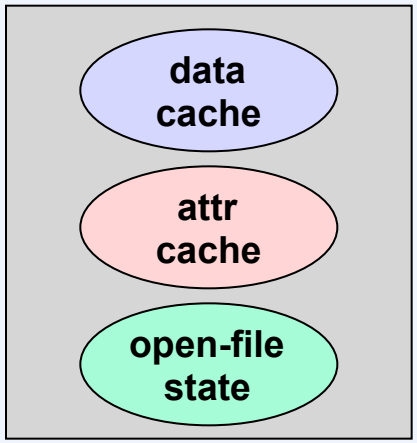
Basic NFS

Extended NFS

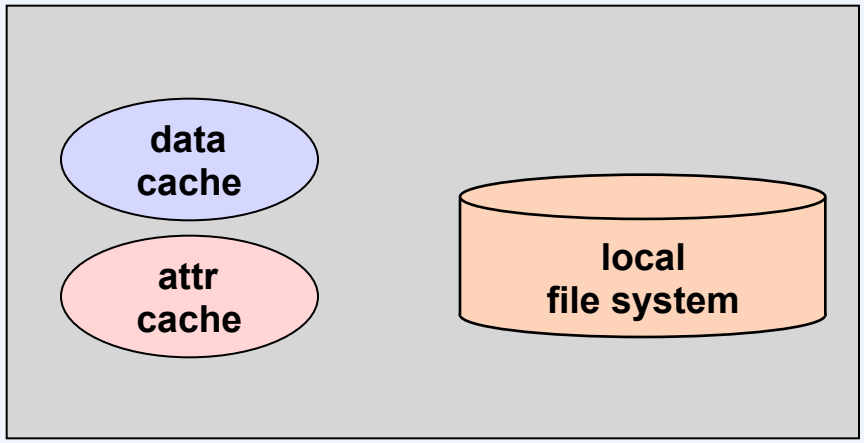
Distribution of Components



NFSv2 client

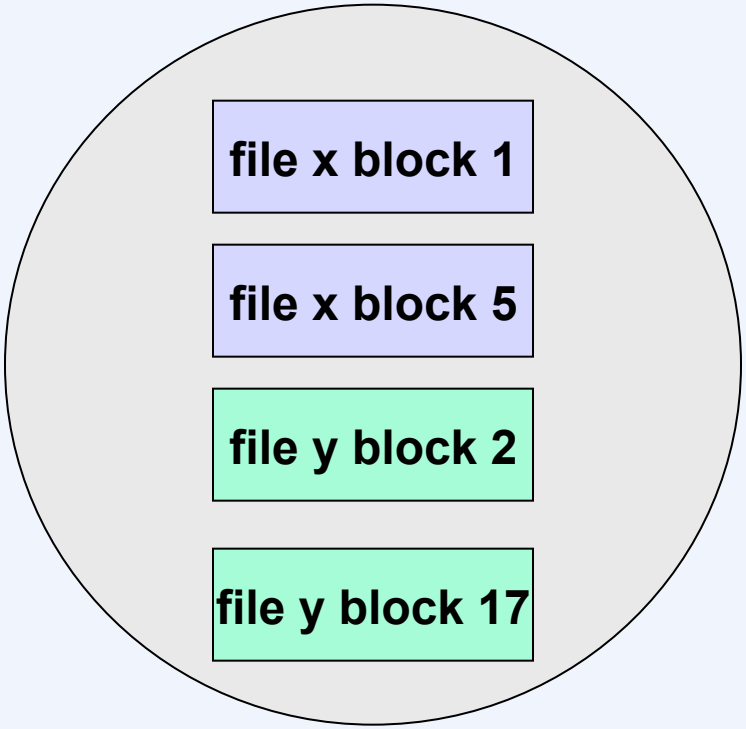


NFSv2 client

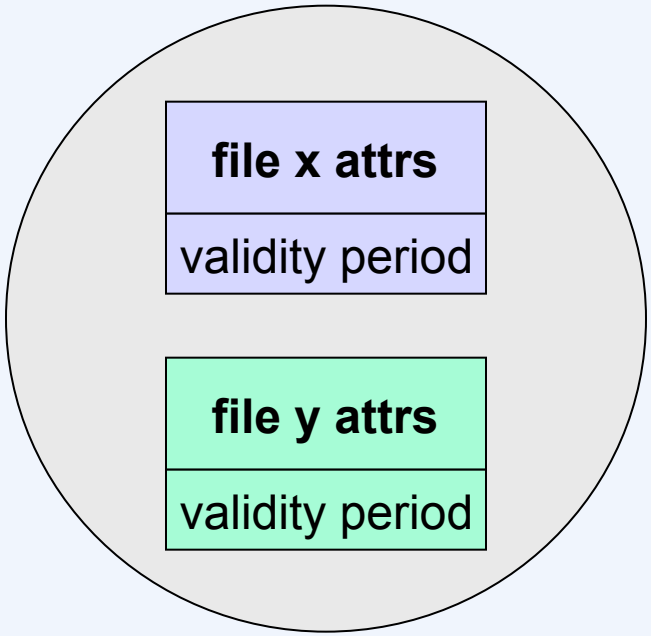


NFSv2 server

Consistency in Basic NFSv2



Data cache

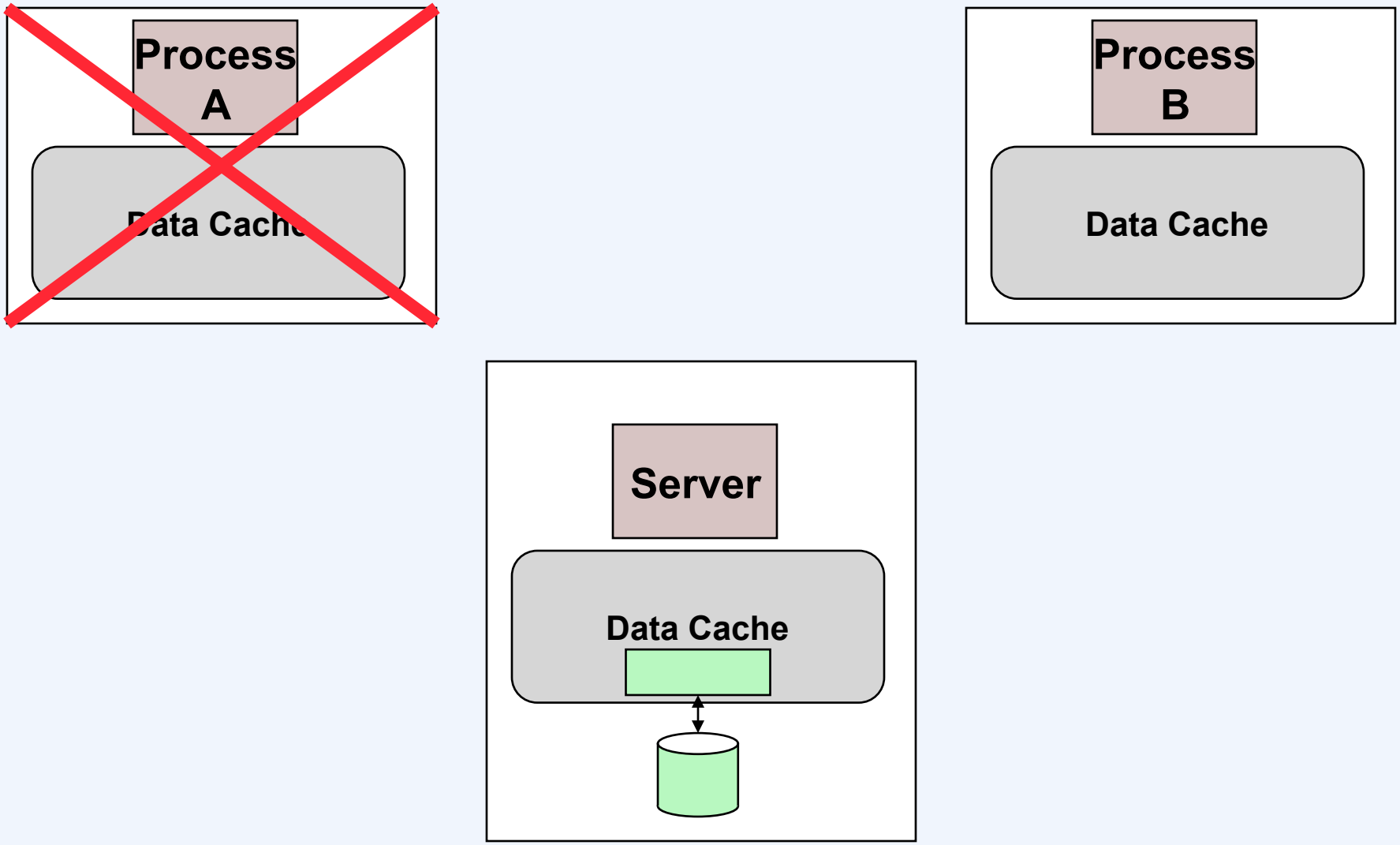


Attribute cache

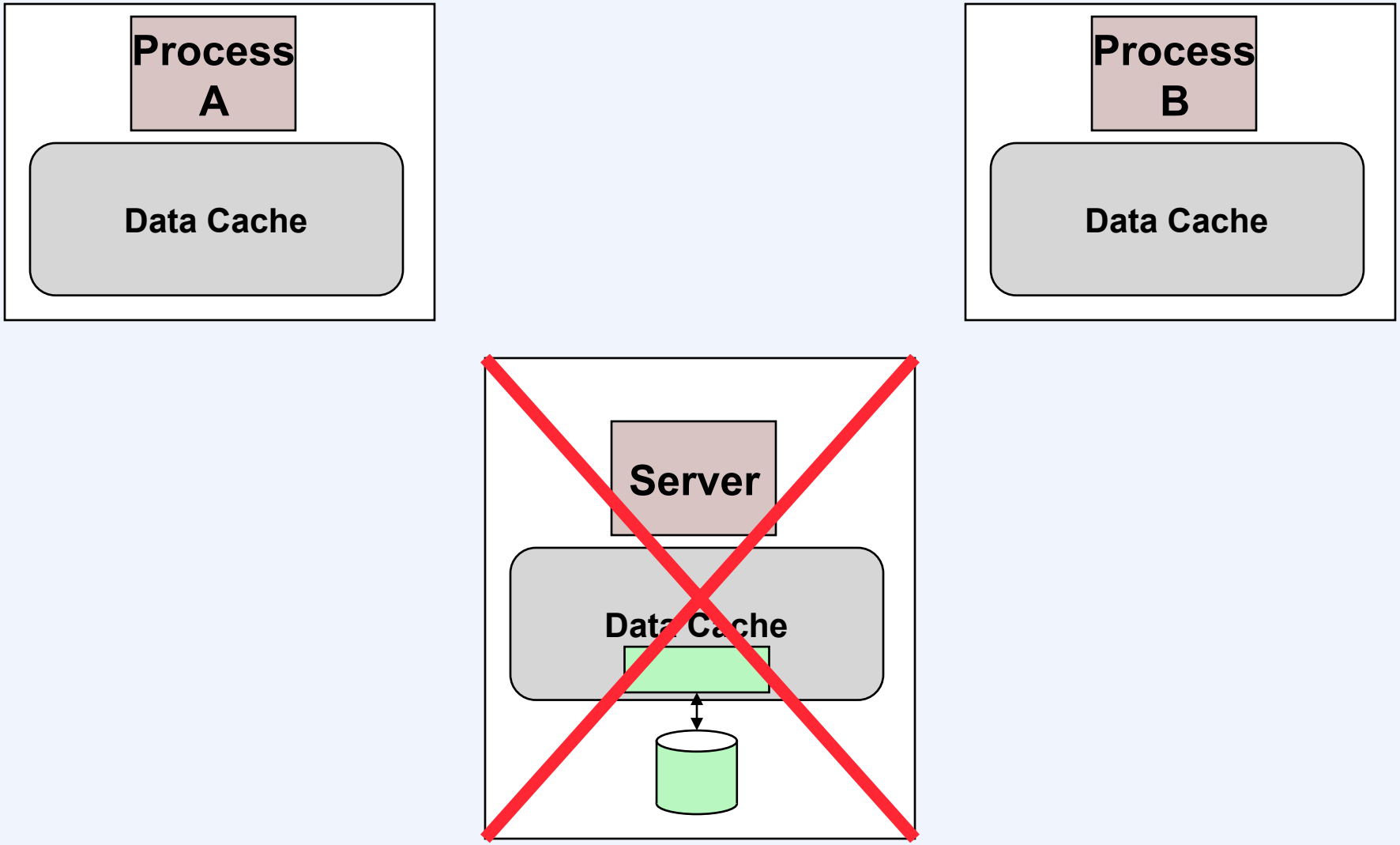
More …

- All write RPC requests must be handled synchronously on the server
- Close-to-Open consistency
  - client writes back all changes on close
  - flushes cached file info on open

Client Crash Recovery



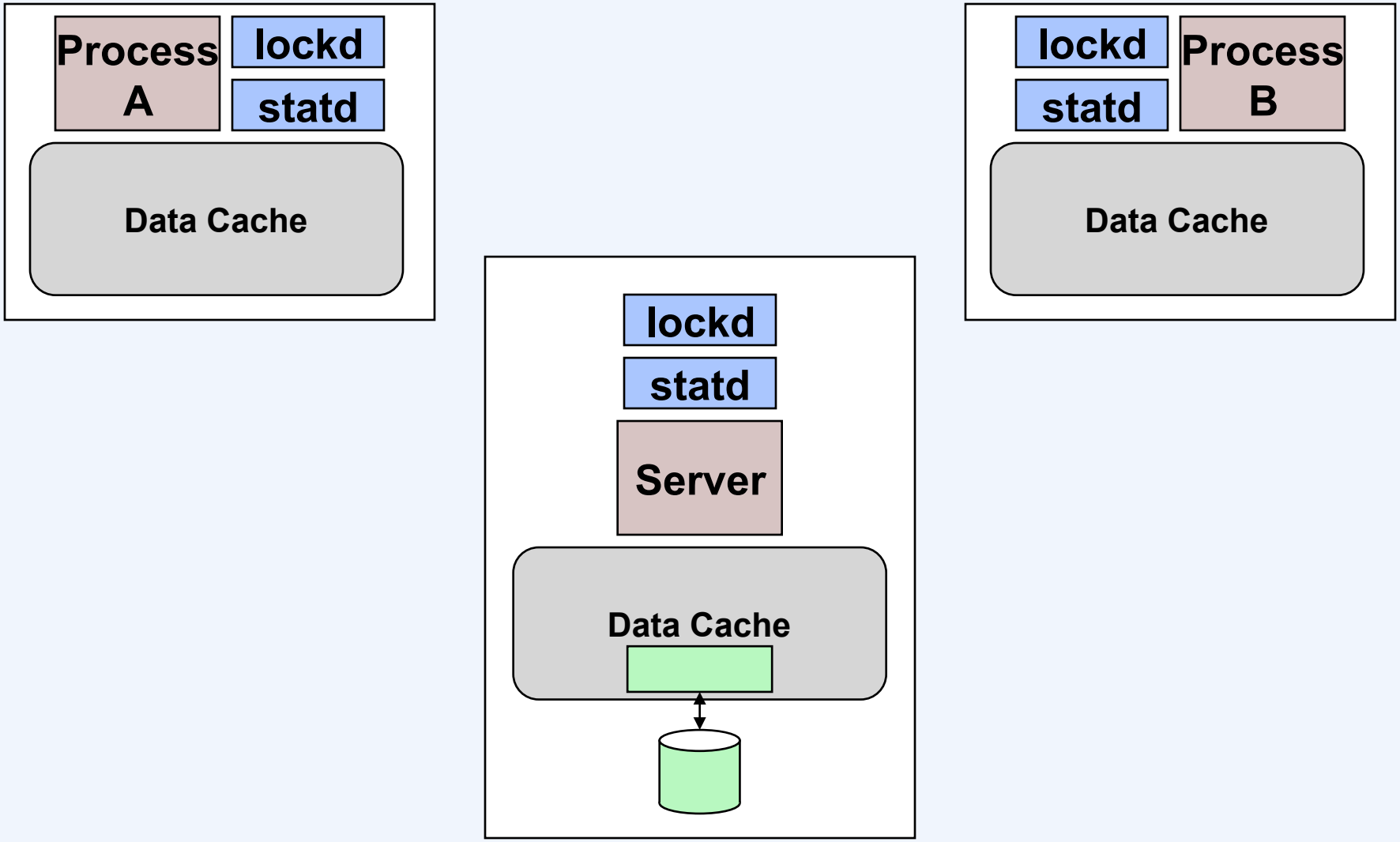
Server Crash Recovery



File Locking

- State is required on the server!
  - recovery must take place in the event of client and server crashes

Network Lock Manager Protocol



NFS Version 3

- In use at Brown and in most of the rest of the world
- Basically the same as NFSv2
  - improved handling of attributes
  - *commit* operation for writes
  - various other things

CIFS

- Common Internet File System
  - Microsoft's distributed file system
- Features
  - batched requests and responses
  - strictly consistent
- Not featured …
  - depends on reliability of transport protocol
  - loss of connection == loss of session

History

- Originally a simple means for sharing files
  - developed by IBM and called server message block protocol (SMB)
  - ran on top of NetBIOS
- Microsoft took over
  - renamed CIFS in late 1990s
  - uses SMB as RPC-like communication protocol
    - runs on NetBIOS
    - usually layered on TCP
    - sometimes no NetBIOS, just TCP

Consistency vs. Performance

- Strict consistency is easy …
  - … if all operations take place on server
  - no client caching
- Performance is good …
  - … if all operations take place on client
  - everything is cached on client
- Put the two together …

  ∅

  - or you can do opportunistic locking

Opportunistic Locks